UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

Estudio y desarrollo de una herramienta en tiempo real para estimulación bidireccional dirigida por codificación temporal en el contexto de peces eléctricos.

Alberto Ayala Valencia
Tutor: Francisco de Borja Rodríguez Ortiz

Julio 2020

Estudio y desarrollo de una herramienta en tiempo real para estimulación bidireccional dirigida por codificación temporal en el contexto de peces eléctricos.

AUTOR: Alberto Ayala Valencia
TUTOR: Francisco de Borja Rodríguez Ortiz

Grupo de Neurocomputación Biológica (GNB)
Dpto. Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Julio de 2020

Resumen

El objetivo de este proyecto es migrar las herramientas de estimulación en ciclo cerrado que habilitan el método experimental conocido como *Temporal code-driven stimulation* (TCDS) a una nueva implementación sobre la API de RTXI, un *framework* actualizado para la realización de experimentos en tiempo real en biología o neurociencia.

TCDS es un método experimental basado en protocolos de estimulación en ciclo cerrado aplicado al estudio de la codificación de información en el sistema nervioso, y que ha sido probado anteriormente en el contexto de la electrorrecepción en peces eléctricos, más precisamente en especímenes de *Gnathonemus Petersii* también conocido como pez elefante.

Esta metodología en ciclo cerrado va a requerir de una implementación en tiempo real estricto, con una frecuencia de trabajo en el orden del sistema real. Las señales emitidas por el sistema biológico de referencia, el pez eléctrico *Gnathonemus Petersii,* contiene pulsos con un tiempo medio de 0.8 ms. La detección y el procesamiento de estos eventos requerirá una frecuencia de trabajo por encima de 5 KHz.

La migración a RTXI aporta muchas ventajas frente a la implementación previa. Primero, integra el método de investigación en un framework mantenido y actualizado como es RTXI. Segundo, adapta la implementación de los módulos a una API estándar que encapsula la complejidad de bajo nivel. Por último, RTXI es un sistema utilizado por distintos grupos de investigación, por lo que esta implementación aporta al método TCDS mayores posibilidades de expansión dentro de la comunidad científica.

El contexto en el que está basado este proyecto, la electrorrecepción, es la capacidad que tienen algunos organismos de utilizar campos eléctricos, normalmente en medios acuáticos, para localizar objetos a su alrededor o comunicarse con otros individuos de su misma especie, por lo que es importante digitalizar la señal de forma que pueda ser tratada mediante algoritmos, aplicando posteriormente una digitalización binaria de la señal en función de si el pez emite o no pulsos eléctricos superiores a un voltaje umbral en una ventana de tiempo definida. Gracias a esto, se podrán realizar estudios de patrones de codificación binaria relevantes para la transmisión de información en el contexto de la Neurociencia Computacional.

Se ha desarrollado la funcionalidad necesaria para la simulación de la emisión de los pulsos eléctricos generados por el pez. Esto es consecuencia directa del periodo de confinamiento por el que hemos pasado a causa del COVID19, por lo que no se pudo experimentar con un pez eléctrico real, tal y como se pensó en un principio.

Además, se ha realizado una serie de pruebas y análisis al final del proyecto en el sistema en tiempo real del Grupo de Neurocomputación Biológica de la Universidad Autónoma de Madrid. Con el objetivo de medir las latencias a las que se ejecutan las tareas en tiempo real de los módulos desarrollados para RTXI. De esta forma se pudo determinar que RTXI proporciona un rendimiento razonable para realizar estudios sobre la codificación de la información en el sistema nervioso.

Por último, también se realizaron experimentos en el mismo sistema en tiempo real del GNB, en los que se montó tanto la parte hardware como software del proyecto en un laboratorio, simulando que un pez eléctrico emitía campo eléctrico mediante dipolos.

Finalmente hay que destacar que los módulos diseñados y desarrollados en este proyecto, implementados en un sistema de preproducción en un entorno virtual (simulación de la

tarjeta de adquisición de datos) se desplegaron con éxito en el sistema de tiempo real situado en el laboratorio del Grupo de Neurocomputación Biológica funcionando correctamente.

Abstract

The objective of this project is to migrate closed-loop stimulation tools that enable the experimental method known as Temporal code-driven stimulation (TCDS) to a new implementation on the RTXI API, an updated framework for conducting real-time experiments in biology or neuroscience.

TCDS is an experimental method based on closed-loop stimulation protocols applied to the study of information coding in the nervous system, and which has been previously tested in the context of electroreception in electric fish, more precisely in specimens of *Gnathonemus Petersii* known as elephant fish.

This closed-loop methodology will require a strict real-time implementation, with a working frequency in the order of the real-time system. The signals emitted by the biological reference system, the electric fish *Gnathonemus Petersii*, contains pulses with an average time of 0.8 ms. Detection and processing of these events will require a working frequency above 5 KHz.

The migration to RTXI have many advantages over the previous implementation. First, it integrates the research method into a maintained and updated framework, RTXI. Secondly, it adapts the implementation of the modules to a standard API that encapsulates low-level complexity. Finally, RTXI is a system used by different research groups, so this implementation gives the TCDS method greater possibilities of expansion in the scientific community.

The context which this project is based on, the electroreception is the ability some organisms have to use electric fields, usually in aquatic environments, to locate objects nearby or communicate with other individuals of the same species, that´s why it´s important to digitize the signal to be processed using algorithms, subsequently applying a binary digitization of the signal depending on whether or not the fish emits electrical pulses above a threshold voltage in a defined time window. Thanks to this, it will be possible to carry out studies of relevant binary coding patterns for the transmission of information in the context of Computational Neuroscience.

The necessary functionality has been developed to simulate the emission of the electrical pulses generated by the fish. This is the direct consequence of the confinement period we have been through because of COVID19, so it was not possible to experiment with a real electric fish, as originally thought.

In addition, a series of tests and analyzes have been carried out at the end of the project in the real-time system of the Biological Neurocomputation Group of the Autonomous University of Madrid. With the aim of measuring the latencies at which the tasks are executed in real time of the modules developed for RTXI. In this way, it was able to determine that RTXI provides a reasonable performance to carry out studies on the encoding of information in the nervous system.

Finally, experiments were also carried out on the GNB real-time system too, in which both the hardware and software parts of the project were assembled in a laboratory, simulating that an electric fish could emit an electric field using dipoles.

Once and for all, it should be noted that the modules designed and developed in this project, implemented in a pre-production system in a virtual environment (simulation of the data acquisition card) were successfully deployed in the real-time system located in the laboratory of the Biological Neurocomputing Group working correctly

Palabras clave

Procesamiento de señal, Sistemas en tiempo real, Electrocomunicación, *Dynamic Clamp*, Secuencias temporales de pulsos, Neurociencia Computacional.

Keywords

Signal processing, Real-time systems, Electrocommunication, Dynamic Clamp, Temporal sequences of pulses, Computational Neuroscience.

Agradecimientos

Me gustaría agradecer tanto a Francisco Rodríguez como a Ángel Lareo la oportunidad de iniciarme en un mundo tan interesante como es el que trata este Trabajo de Fin de Grado. Agradecer también a los que considero mis dos tutores por todo el apoyo que me han brindado, las ganas y el esfuerzo que han puesto para hacer juntos este proyecto.

Agradecer también a mi familia por ser un pilar muy importante en mi vida porque siempre están cuando se les necesita.

Por último, agradecer a todos mis compañeros y amigos tanto de la universidad como externos por haber compartido estos 4 años de mi vida.

INDICE DE CONTENIDOS

INDICE DE FIGURAS

1 Introducción

El método experimental *Temporal code-driven stimulation* (TCDS) [1, 2] define un protocolo de estimulación basado en la detección de códigos binarios en señales biológicas. Una de las principales características de este método es la utilización de una metodología denominada **estimulación de ciclo cerrado**, mediante la cual se establece una comunicación bidireccional con el sistema biológico. Para ello, se monitoriza una señal biológica, y en función de esta actividad, se estimula al organismo que se desea estudiar dando lugar a una influencia mutua. Por otro lado, existen metodologías tradicionales de ciclo abierto donde no se establece este proceso bidireccional, sino que simplemente se estimula al organismo mediante señales pregrabadas en tiempos predeterminados monitorizando a continuación la respuesta emitida por el sistema biológico ante estos estímulos. Esta última metodología tiene claras limitaciones, no pudiendo cambiar el tipo de estimulación en función del comportamiento del sistema que se analiza, permaneciendo ocultos los procesos de análisis de información que solo tienen lugar si existe una comunicación bidireccional. Es por esto por lo que se opta por un estudio con metodologías de ciclo cerrado [3, 4, 5, 2, 6, 7, 8].

El uso de este tipo de metodologías requiere de operaciones que se realizan en el orden de los milisegundos, en concreto, TCDS ha sido probado en el ámbito de la electrorrecepción en peces eléctricos. La especie utilizada como referencia en este proyecto, *Gnathonemus Petersii*, emite pulsos eléctricos con un tiempo medio de 0.8 ms. Además, estas operaciones deben realizarse con una robustez temporal, es decir, con mínima latencia en los tiempos de ejecución, es por esto por lo que este proyecto se ha desarrollado en un entorno basado en un sistema en tiempo real.

Un sistema en tiempo real es aquel que interacciona con procesos físicos que ocurren en el mundo real y respondiendo a los estímulos que se producen en un tiempo determinado.

La implementación original de TCDS desarrollada por Lareo [1, 2], utiliza la interfaz de experimentos **ADClamp** (ver apéndice F) sobre **RTAI** (ver apéndice D). En el marco de este trabajo se ha portado la funcionalidad de este método a **RTXI** (ver sección 1.3), una interfaz actualizada diseñada específicamente para la realización de experimentos en tiempo real y utilizado en el campo de la investigación en neurociencia y biología.

RTXI aporta un gran número de ventajas en comparación con otras herramientas en tiempo real, como la facilidad de instalación y de creación de módulos propios, o la encapsulación de los módulos en una API estandarizada.

1.1 Metodologías de control y Dynamic Clamp

Existen dos tipos de operaciones utilizadas en sistemas de control, operaciones en ciclo cerrado y operaciones en ciclo abierto. La diferencia entre ambas está en la retroalimentación usando la salida obtenida. Un sistema controlado por operaciones en ciclo abierto solo actúa sobre la base de la entrada, la salida no tiene ningún efecto en la acción de control; sin embargo, en un sistema controlado por operaciones en ciclo cerrado existe una retroalimentación, es decir, se analiza la salida actual y actúa en función de la

salida obtenida. En este proyecto se usa una técnica conocida como estimulación en ciclo cerrado que consiste en estimular un sistema biológico en función de su actividad [3, 4].

Durante el análisis de una señal biológica, como puede ser la señal en forma de pulsos generada por el *Gnathonemus Petersii,* o la señal eléctrica generada por una neurona, se monitoriza en tiempo real los pulsos eléctricos como entrada del sistema de control y, en función de la señal recibida se estimula al sistema, que reacciona a esa estimulación modificando su señal, estableciéndose así una estimulación bidireccional. TCDS está basado en otra técnica de estimulación en ciclo cerrado y en tiempo real conocida como ***Dynamic clamp***. Este método electrofisiológico consiste en estimular células que pertenecen a una red neuronal en función del potencial de sus membranas mediante un proceso bidireccional de monitorización con el objetivo de estudiar su comportamiento y cómo se comunican entre ellas [9, 10, 5, 11].

El concepto de control en ciclo cerrado o *Dynamic clamp* se puede generalizar tal y como se analiza en el estado del arte (ver sección 3), esta técnica se usa dentro de una diversidad considerable de ámbitos, como puede ser el estudio y control de las cardiopatías, los trastornos neurológicos y neuropsiquiátricos, sistemas de comunicación, estudio de enfermedades del ser humano reproducibles en otros sistemas biológicos, aplicaciones en el campo de la robótica entre otros [12-16].

1.2 Sistemas en tiempo real

El uso de técnicas basadas en *Dynamic clamp* requiere que las operaciones de control y estimulación se realicen en intervalos de tiempo relativamente cortos y de forma consistente, además, el *Gnathonemus Petersii* emite señales de campo eléctrico en este mismo orden. Es por este motivo por lo que todas las funcionalidades desarrolladas en este proyecto se han implementado en RTXI, una interfaz de experimentos que trabaja sobre un sistema operativo en tiempo real (Xenomai).

Un sistema en tiempo real tiene que cumplir con las siguientes características:

- **Determinista**, es decir, el sistema tiene que conocer con exactitud el tiempo que tarda en responder a una interrupción o petición de servicio.
- **Responsividad**, esta cualidad se enfoca al tiempo de demora que hay entre que se atendió a una interrupción y el momento en el que se inicia la tarea, y al tiempo que tarda en realizarse dicha tarea [1].
- En un sistema operativo en tiempo real el usuario tiene la capacidad de establecer la **prioridad** de una tarea por encima de otros procesos del SO.
- Además, debe ser un sistema **estable**, cumpliendo las tareas con mayor prioridad cuando no se puedan cumplir los límites de tiempo que se le imponen preservando la mayor cantidad de datos y servicios del sistema posibles.
- Por último, un sistema en tiempo real debe ser **confiable**, es decir, debe prestar una calidad de servicio óptima, así como un correcto funcionamiento ante fallos y situaciones límite.

Para que un sistema sea considerado de tiempo real, debe tener un tiempo máximo conocido para cada una de las tareas que realiza. Debido a esta condición, existen dos

tipos de sistemas que funcionan en tiempo real en base a la flexibilidad con la que realizan las tareas.

Los sistemas ***hard*** *real-time* son aquellos que garantizan de manera estricta un requerimiento temporal donde un incremento de latencia se considera un error del sistema. El airbag de un coche es un ejemplo de sistema *hard real-time*. Los sistemas ***soft*** *real-time* son aquellos en los que los requerimientos *real-time* no son tan estrictos, como por ejemplo el sistema de audio en un ordenador [17].

Como el entorno en el que se realiza este trabajo requiere una estimulación bidireccional en un orden de milisegundos se necesita hacer uso de un sistema en tiempo real. Para este TFG se ha usado RTXI basado en **Xenomai** (ver apéndice E**)** para facilitar el estudio y el análisis de la codificación de información en el sistema nervioso en los peces eléctricos.

1.3 Interfaz de eXperimentos en Tiempo Real (RTXI)

RTXI (*Real-Time eXperiment Interface*) [18] es una herramienta de adquisición, control y tratamiento de datos en tiempo real dedicada específicamente a la investigación en sistemas biológicos. Esta tecnología usa un sistema operativo en tiempo real basado en Linux, ya que trabaja utilizando su núcleo a la hora de dar prioridad a los procesos que están teniendo lugar.

RTXI es un sistema rápido, modular y de código abierto usado en laboratorios de todo el mundo para la investigación en sistemas biológicos.

La adquisición de los datos y las interfaces analógicas o digitales se realizan en tiempo real a través de la interfaz de controlador **Analogy** (ver apéndice C) ya que proporciona soporte para una gran cantidad de **tarjetas de adquisición de datos** (ver apéndice A).

Las principales características de esta herramienta son:

- La arquitectura modular permite múltiples instancias de los módulos desarrollados por el usuario facilitando la reutilización, organización y depurado de los algoritmos desarrollados.
- El sistema de adquisición de datos permite la transmisión de múltiples canales de datos junto con metadatos.
- La herramienta es compatible con una gran variedad de tarjetas DAQ y hardware externo.
- Durante la ejecución es posible cambiar los parámetros experimentales de forma muy sencilla mediante una interfaz gráfica sin recompilar el módulo ni detener el proceso en tiempo real.
- Además, es capaz de guardar y volver a cargar todo el entorno de trabajo.
- Existe una clase base que permite al usuario construir módulos con una interfaz gráfica de usuario personalizable.
- Contiene herramientas que facilitan la medida de latencias en tiempo real.
- Se pueden conectar y sincronizar las entradas y salidas entre módulos.
- Se pueden capturar y grabar en disco los datos obtenidos durante una ejecución en tiempo real.

La arquitectura de esta herramienta como se puede ver en la Figura 1 se puede dividir en dos partes, por un lado, el tiempo real y, por otro lado, la interfaz o espacio para el usuario que no está en tiempo real.

RTXI cuanta con módulos que son propios del sistema y con una API para que los usuarios puedan implementar nuevos módulos. RTXI cuenta con la herramienta Conector que encapsula la comunicación entre módulos de manera transparente para el desarrollador. De esta forma, los módulos son capaces de comunicarse entre ellos mediante un sistema de señales que admite la programación síncrona y el tratamiento de eventos asíncronos.

El proceso de ejecución de la herramienta es el siguiente, el proceso en tiempo real indica a cada controlador de tarjetas DAQ que obtenga todas las entradas activas del hardware externo de las que hace uso el estudio que se está llevando a cabo, estos datos se ponen a disposición de otros componentes mientras se ejecuta cualquier instrucción en tiempo real programada por los módulos que se están utilizando (cargados). Cuando termina la ejecución de estos módulos se pide a cada controlador de salida de tarjeta DAQ que escriba la señal en los dispositivos hardware del estudio.

Por otro lado, se ejecuta el proceso encargado de administrar la interfaz de usuario, así como los eventos sucedidos a través de esta.

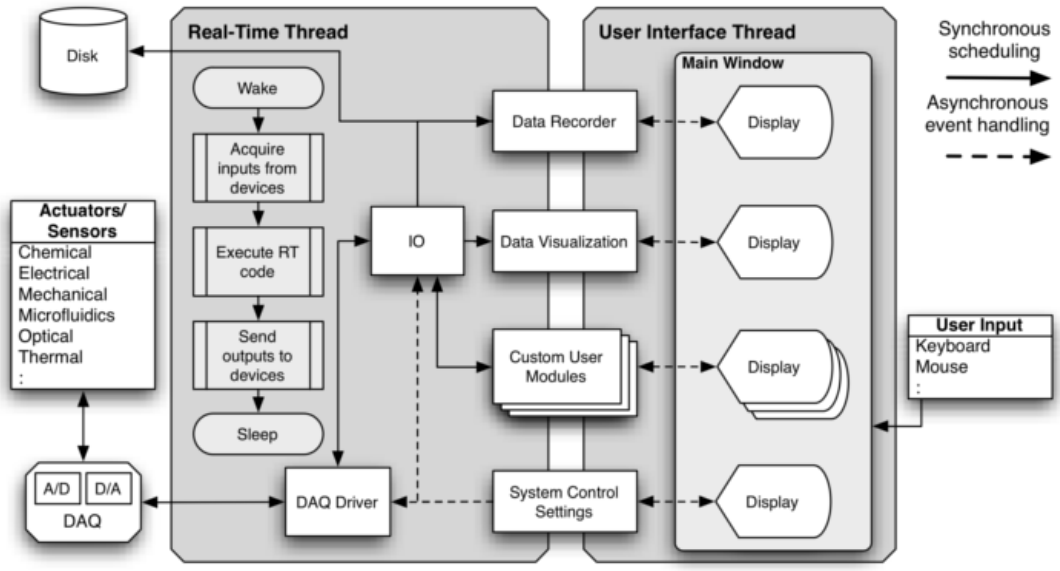

Figura 1 Arquitectura de RTXI. Figura extraída de [44].

Además, los módulos se compilan fuera del árbol propio de RTXI haciendo que la estructura tenga las siguientes características diferenciadas:

- Se reducen los gastos que maximizan el rendimiento en tiempo real cuando se carga el mínimo número de módulos iniciales.
- Reutilización de código.

- Facilidad para compartir módulos creados por los usuarios distribuyendo el código fuente o la versión compilada.
- Se puede cargar, guardar y configurar el espacio de trabajo en la herramienta siguiendo las especificaciones establecidas por el usuario.
- Ampliación de funcionalidad sin cambios en el código fuente de RTXI.

Tanto ADClamp basado en RTAI [1, 2] como RTXI basado en Xenomai ofrecen interfaces de experimentación en tiempo real robustas y de muy bajas latencias que pueden ser utilizadas para implementar TCDS.

No obstante, cabe destacar varias diferencias entre ambas. ADClamp es una interfaz de experimentos que actualmente está desactualizada ya que usa librerías que están en desuso. A su vez, RTXI es una herramienta con la que hoy en día trabajan un gran número de grupos de investigación científica [11] posibilitando a esta comunidad el uso de esta herramienta para el diseño de sus experimentos. Además, RTXI sigue recibiendo soporte software y su instalación es bastante sencilla y práctica mientras que la instalación de ADClamp es más compleja. Por otro lado, RTXI es una herramienta que tiene como principal enfoque la investigación biológica, por lo tanto, todos los módulos que contiene el sistema por defecto están diseñados para el desarrollo de proyectos en este campo. Otra ventaja que RTXI aporta al proyecto es la posibilidad de encapsular funcionalidad con facilidad, se pueden dividir las distintas tareas en tiempo real en módulos y con el tiempo ampliar su funcionalidad, añadir nuevos módulos, etc.

Como ya se ha mencionado en esta sección, la comunicación entre procesos es mucho más sencilla, con el módulo del sistema Conector (ver sección 4.1.1), RTXI se encarga de conectar las salidas y entradas de los módulos que se hayan desarrollado, así como con otros módulos del sistema haciendo muy cómoda la comunicación entre procesos en tiempo real. En cuanto a la comunicación con la interfaz (usuario) RTXI también se encarga de gestionarlo, el usuario puede variar los parámetros introducidos en el experimento y RTXI se encargará de comunicárselo a las tareas que se estén ejecutando en tiempo real, estas tareas reciben la señal, cambian los parámetros y actúan siguiendo los pasos que se le hayan especificado. En ambos casos RTXI sincroniza el intercambio de información de forma asíncrona y siguiendo las especificaciones predefinidas en el experimento.

Por último, otra de las principales ventajas que aporta la modularidad es facilitar el análisis de fallos en la implementación. Esta ventaja es consecuencia de la encapsulación, ya que organiza el experimento en tareas y permite localizar y corregir el fallo con mucha más precisión y rapidez.

2 Motivación

La principal motivación es contribuir a desarrollar métodos de análisis de la codificación de información en el sistema nervioso, adaptando y añadiendo nueva funcionalidad al método TCDS desarrollado por Lareo [1, 2].

En la actualidad, existe un gran número de investigaciones biológicas destinadas a distintos fines, pero con un objetivo predominante, conocer mejor como funciona los mecanismos neurológicos biológicos y cómo podemos aplicar este conocimiento para favorecer la vida humana. Existen ya algunas aplicaciones fruto de estas investigaciones que ayudan a muchas personas con patologías cardiovasculares y neurológicas. También, se está investigando la recuperación de la sensibilidad en zonas donde se han perdido las terminaciones nerviosas e incluso aplicaciones robóticas que pueden ser muy beneficiosas es un gran abanico de ámbitos [12-16].

Este proyecto pretende ser parte de estas investigaciones y contribuir en ellas mediante la migración a un sistema actualizado de una herramienta para realizar estudios relacionados con el procesamiento de la información del sistema nervioso de los peces eléctricos, haciendo uso de técnicas en ciclo cerrado y en tiempo real.

Los módulos desarrollados en este proyecto se publican bajo licencia libre para su uso y modificación, pueden encontrarse en https://github.com/GNB-UAM/tcds-rtxi.

2.1 Objetivos

El principal objetivo de este proyecto es desarrollar una aplicación con fines sobre todo de investigación, para el análisis de la codificación de la información en el sistema nervioso de los peces eléctricos mediante el uso de técnicas en ciclo cerrado y en tiempo real. Dada la definición principal del proyecto en el desarrollo de éste se pretenden alcanzar los siguientes subobjetivos.

- Simulación: Como ya se ha mencionado anteriormente, durante el desarrollo de este trabajo hemos atravesado un periodo de confinamiento por lo que era imposible probar la funcionalidad implementada con un pez eléctrico. Debido a estos motivos se ha desarrollado una parte de la funcionalidad de la aplicación para la simulación de la señal emitida por el pez, de tal forma que se puede usar para comprobar el correcto funcionamiento del protocolo desarrollado, la estimulación bidireccional, así como los futuros protocolos que se vayan desarrollando, haciendo uso de esta herramienta.
- Ciclo cerrado: Este proyecto pretende ser la base de fututos experimentos y estudios en los que se analice el comportamiento de los peces eléctricos frente a estímulos a los que se ven sometidos, es decir, se establece una comunicación bidireccional con el pez, este tipo de metodologías se conoce como estimulación en ciclo cerrado y proporciona información de gran utilidad que otros tipos de metodologías no ofrecen. Por lo tanto, en el proyecto se desarrollará funcionalidad que permita el estudio del procesamiento de información en este tipo de sistemas biológicos usando metodologías de ciclo cerrado, desde la adquisición de datos, pasando por el

procesado y análisis de la información y terminando por la estimulación al organismo.

- Generación del protocolo: Existe una amplia diversidad en cuanto a la forma en la que se puede comunicar un sistema artificial con un sistema biológico. Este proyecto pretende ser una base escalable por lo que se implementará un protocolo (el protocolo TCDS) de comunicación bidireccional con el pez que permita hacer experimentos y obtener datos que tras su análisis sean de utilidad y permita el estudio del comportamiento del organismo cuando se enfrenta a distintas situaciones.
- Digitalización binaria de la señal: Una de las partes principales de este proyecto es convertir el campo eléctrico emitido por el pez en una señal digital que se pueda interpretar, para ello se dividirá la señal en trozos en función de una variable temporal y se codificará la señal en unos y ceros en función de si se detecta voltaje o no por encima de un umbral en estas divisiones. Este proceso se explica con más detenimiento en la metodología del proyecto.
- Comprobación del protocolo: Como es lógico, es importante comprobar que toda la funcionalidad desarrollada en el proyecto desempeña su cometido correctamente. Por este motivo se va a comprobar de dos formas:
  - ❖ Depurado: Comprobación de que se produce la estimulación en el momento preciso siguiendo el flujo establecido por el protocolo implementado. Se comprobará que las latencias a las que se estimula el proyecto están dentro de lo esperado y que no influyen a la hora de comunicarse con el pez.
  - ❖ Experimental: Mediante la comunicación directa con un hardware que simule al sistema biológico comprobar que el sistema artificial reaccione en función de la señal recibida y se lancen estímulos validando que todo el sistema de simulación realizado funciona correctamente.

2.2 Organización de la memoria

La memoria consta de los siguientes capítulos con sus respectivas subsecciones:

- **Capítulo 1: Introducción.**
- **Capítulo 2: Motivación.**
- **Capítulo 3: Estado del arte.**
- **Capítulo 4: Metodología.**
- **Capítulo 5: Análisis y diseño de la aplicación.**
- **Capítulo 6: Desarrollo.**
- **Capítulo 7: Integración, pruebas y resultados.**
- **Capítulo 8: Conclusiones y trabajo futuro.**

3 Estado del arte

El uso de técnicas de estimulación en ciclo cerrado ha evolucionado con mayor notoriedad desde los años 90 hasta nuestros días. En la actualidad, existen diversas herramientas que permiten el desarrollo de experimentos mediante el uso de metodologías *Dynamic Clamp* y con las que se podría haber trabajado para llevar a cabo este proyecto [9, 10, 19].

Un claro ejemplo de este tipo de herramientas es ***Extended Dynamic Clamp***, este programa está desarrollado en Windows y se basa en la lectura del reloj interno en la tarjeta de adquisición de datos para trabajar en tiempo real en un sistema operativo que no lo es. Sin embargo, este sistema solo funciona correctamente cuando la carga computacional es relativamente baja. Además, la implementación de modelos suele ser bastante compleja de realizar [5].

Otra de las herramientas que trabajan con este tipo de metodologías es **RTLDC**, este programa funciona sobre Linux en tiempo real, es flexible y fácil de usar gracias a una interfaz gráfica intuitiva, cuenta con una gran velocidad, bajo coste computacional y un buen rendimiento, además permite la carga de modelos y la modificación de parámetros en línea. Sin embargo, carece de herramientas que permitan analizar datos, así como de un registro donde se pueda ver lo que ha sucedido [5].

MRCI es otro ejemplo de software que también funciona en Linux en tiempo real que aporta una gran cantidad de ventajas como secuencias de comandos para implementar protocolos o la capacidad de registrar datos, aunque no dispone de una interfaz gráfica [20].

En el mundo de la investigación usando este tipo de metodologías existen muchas otras herramientas que son usadas en una gran cantidad de proyectos distintos, como ***LabVIEW-RT Dynamic Clamp***, ***G-clamp***, ***QuB*** [21].

En cuanto a los avances en los que se está trabajando actualmente existe un gran abanico de posibilidades donde esta tecnología está adquiriendo cada vez más importancia. Por un lado, la estimulación en ciclo cerrado está siendo de gran importancia en el campo de la estimulación cerebral no invasiva, de esta forma, se puede monitorizar la actividad cerebral y estimular en función de ella para la supresión de la actividad patológica de trastornos neurológicos y neuropsiquiátricos como el Parkinson, la epilepsia y trastornos obsesivo-compulsivos [12-16].

Por otro lado, este tipo de metodologías están contribuyendo notoriamente en el mundo de las cardiopatías. Existen herramientas de estimulación en ciclo cerrado integradas en marcapasos capaces de monitorizar y actualizarse en función de los estímulos fisiológicos recibidos por el cuerpo, capaces de evitar, por ejemplo, síncopes vasovagales [22].

En lo que concierne al ámbito de este proyecto, existen herramientas desarrolladas que se centran en la investigación. La mayoría de estos proyectos son herramientas capaces de interactuar con neuronas mediante estimulación en ciclo cerrado y analizar los resultados obtenidos de esta interacción.

Por ejemplo, se han desarrollado herramientas de código abierto que usan este principio para determinar la relación causal entre la actividad neuronal y el comportamiento animal [23, 24].

Investigadores como Ricardo Pineda, Christine E. Beattie y Charles W. Hall Han desarrollado una herramienta cuyo objetivo es determinar si una plataforma híbrida en la que se estimulará la activación del tronco encefálico en respuesta a una inminente actividad convulsiva podría prevenir dichas convulsiones en personas con epilepsia. Al igual que en este proyecto, se eligió a un animal para el experimento, el pez cebra ya que es capaz de reproducir muchas enfermedades humanas y es fácil de monitorizar [25].

En cuanto a peces eléctricos, uno de los peces con los que más se investiga es el ***apteronotus leptorhynchus***. Por ejemplo, en su tesis, Maciver investiga como trabajan el cuerpo y el sistema nervioso de este pez en el proceso de adquisición sensorial mediante estimulación en ciclo cerrado [26].

Gerri Mileva, Daniel Zysman, Sally Groothuis y John E Lewis también realizaron un estudio muy similar a este proyecto donde investigaban de forma in vitro y en estimulación en ciclo cerrado mediante una red híbrida cómo es el sistema nervioso de este pez [27].

También se han realizado múltiples estudios relacionados con la electrocumunicación y la electrolocalización activa como el de Caroline G. Forlim y Reynaldo D. Pinto [28], donde desarrollaron una herramienta que permite la estimulación realista automática en tiempo real y la grabación de los pulsos eléctricos. De esta forma se puede estudiar la codificación binaria de los IPI emitidos por el pez y evaluar los datos obtenidos en función de su comportamiento. Otro estudio puntero y estrechamente relacionado con la electrorrecepción activa es el desarrollado por los investigadores anteriores junto con Francisco B. Rodríguez [6], donde aplican un protocolo dependiente de actividad en tiempo real que usan para estudiar cómo procesan los peces débilmente eléctricos sus propias señales eléctricas, más concretamente el *Gnathonemus Petersii*

Todos estos proyectos de investigación parten con una premisa común, conocer más sobre cómo funciona el sistema nervioso y más específicamente en el contexto de los peces eléctricos de descarga débil. Gracias a estos proyectos se pueden tratar enfermedades como la epilepsia, monitorizar patologías cardiacas, estimular médulas dañadas para la recuperación de zonas sensibles perdidas, etc.

Por último, otro de los estudios más actuales en los que se analiza esta metodología y principal referente de este proyecto es el estudio realizado por Lareo [1, 2], donde se estudia la estimulación en ciclo cerrado basada en códigos, la influencia del retraso pulso-estímulo o la relevancia de la memoria corto plazo para el procesamiento de la información en el *Gnathonemus Petersii*, coloquialmente conocido como pez elefante, con el objetivo de conocer más sobre el funcionamiento del sistema nervioso y sus posibles futuras aplicaciones.

El uso de peces eléctricos para el estudio del sistema nervioso y el desarrollo de nuevos métodos con aplicación en neurociencia siguen desarrollándose. Este TFG nace de la idea de continuar con esta investigación y contribuir aportando una herramienta más

actualizada y con mayor capacidad de difusión que implemente el método TCDS de análisis de codificación de información en el sistema nervioso.

De todas las interfaces de experimentación en tiempo real existentes con las que se podía haber desarrollado esta aplicación se opta por RTXI porque es la que proporciona un mayor número de ventajas. La principal ventaja que aporta RTXI es una interfaz actualizada ampliamente utilizada para la experimentación en biología y neurociencia. Cuenta con módulos de sistema específicamente preparados para su uso en estudios y tareas experimentales en tiempo real, lo que habilita la realización de ciclos cerrados de estimulación. Actualmente se usa en más de 60 laboratorios distribuidos por todo el mundo y que ha dado lugar a una gran cantidad de artículos de investigación [11]. Además, RTXI aporta modularidad, escalabilidad, depurado, interfaz gráfica intuitiva, facilidad en la construcción de funcionalidad, una arquitectura que facilita la ejecución de todos sus procesos tanto tiempo real como espacio de usuario, etc. Todas estas características hacen de esta herramienta la mejor opción para este trabajo.

4 Metodología

La metodología de trabajo de este proyecto consiste en utilizar la herramienta en tiempo real RTXI para portar el método TCDS.

RTXI (*the Real-Time eXperiment Interface*) es una herramienta de adquisición y control de datos en tiempo real diseñada específicamente para estudios e investigaciones biológicas en la que se desarrollará este proyecto.

Para que la señal de campo eléctrico emitida por el pez pueda ser utilizada por el protocolo experimental se llevará un proceso de codificación binaria explicado a continuación, esta señal se digitalizará de forma binaria en palabras de la misma longitud, cada una de estas palabras estará formada por un conjunto de bits cuyo valor será 0 ó 1 en función de si se ha detectado un pulso por encima de un umbral en una franja de tiempo determinada.

4.1 Interfaz de eXperimentos en Tiempo Real (RTXI)

Una parte importante de esta sección es conocer las utilidades que aportan los módulos de RTXI.

4.1.1 Módulos del Sistema

RTXI implementa una serie de funcionalidades base predefinidas y disponibles, enfocadas para la realización de experimentos biológicos y clave para el desarrollo de este tipo de proyectos.

Panel de control

Desde este módulo podemos configurar de forma general y para todos los canales de entrada los parámetros más importantes del proyecto.

Se pueden configurar las tarjetas de adquisición de datos que se van a utilizar en el proyecto tanto de entrada como de salida, así como los canales analógicos de los que disponen estas tarjetas y parámetros como el rango, la escala y el offset.

Además, en este módulo es donde se establece la frecuencia o período con el que se van a llevar a cabo todas las tareas en tiempo real.

Osciloscopio

Este módulo permite trazar cualquier tipo de señal que se le especifique, como pueden ser las recogidas por las DAQ. También puede recoger cualquier tipo de señal de los módulos desarrollados por los usuarios, ya sean entradas, salidas e incluso cómo varían los parámetros establecidos dentro de estos módulos.

El osciloscopio permite modificar la traza de las señales ya sea la escala o el offset a gusto del desarrollador e incluso configurar un disparador para que cuando una señal cumpla ciertos requisitos el osciloscopio se congele.

Data Recorder

Con este módulo se pueden recoger los datos obtenidos como entrada, salida e incluso parámetros de los módulos desarrollados por los usuarios y guardarlos en disco con un formato **HDF5** (ver apéndice B). Durante la ejecución del experimento en tiempo real este módulo se encarga de recopilar la información especificada y escribirla estructuradamente junto con metadatos que son de gran utilidad.

Conector

La funcionalidad de este módulo es de gran importancia para el desarrollo de este tipo de experimentos, ya que permite la conexión entre módulos y la conexión de estos con las DAQ.

Este módulo permite el flujo de datos que tiene como origen las tarjetas de adquisición de datos o la salida de cualquier módulo y como destino la entrada de cualquier otro módulo.

RT Benchmarks

Este módulo se encarga de obtener y mostrar las estadísticas en tiempo real de RTXI durante la ejecución del experimento. Para que un estudio se ejecute correctamente todas las tareas que participan deben realizarse dentro del periodo establecido por el sistema. Con este módulo podemos analizar en seguimiento continuo el tiempo necesario para completarlas, así como el periodo real en el que se ejecutan. Este módulo también muestra información sobre el tiempo de cálculo total y el periodo del peor caso, así como el *jitter*.

4.1.2 Módulos de usuario

Una de las ventajas con las que cuenta RTXI es la facilidad con la que un usuario puede desarrollar sus propios módulos pudiendo encapsular toda la implementación del proyecto por funcionalidades y la posibilidad de conexión entre ellas.

Además, RTXI dispone de una plantilla base que los desarrolladores pueden usar para implementar sus módulos personalizados de forma muy simple y práctica (para más detalles ver apéndice G).

Otra de las grandes ventajas que tiene RTXI es que al ser una herramienta de código abierto puedes acceder e instalar módulos que hayan sido desarrollados por otros usuarios y que pueden ayudar y contribuir con el experimento que estés realizando.

En este proyecto se han desarrollado 3 módulos, el generador de señales pregrabadas, el *WORDS* y el Generador de estímulos, que son los encargados de desarrollar la funcionalidad del proyecto y que se explican detalladamente en el diseño y desarrollo de este (ver sección 6).

4.2 Codificación binaria del campo eléctrico

Como ya se ha mencionado en la introducción para el estudio del comportamiento del sistema biológico de este proyecto se ha digitalizado de forma binaria la señal eléctrica generada por el pez. Posteriormente se divide en trozos, en función de un número predefinido de bits dando lugar a palabras [1, 2].

Para recodificar la señal a una codificación binaria se puede definir el bit asociado a cada ventana de tiempo (con una duración predefinida) en función de si se detecta o no un pulso eléctrico, es decir, la señal resultante se codifica de manera binaria en función de si el pez emite o no un pulso en intervalos de tiempo predefinidos. Si se detecta un pulso, el bit pasa a valer 1, por el contrario, si durante toda la ventana de tiempo el pez no emite ningún pulso, el bit pasa a valer 0.

La longitud temporal que da lugar a la digitalización de la señal se la conoce como tiempo de bin. Un número de bits predefinido da lugar al tamaño del código, un ejemplo de esta estructura lo podemos ver en la Figura 2.

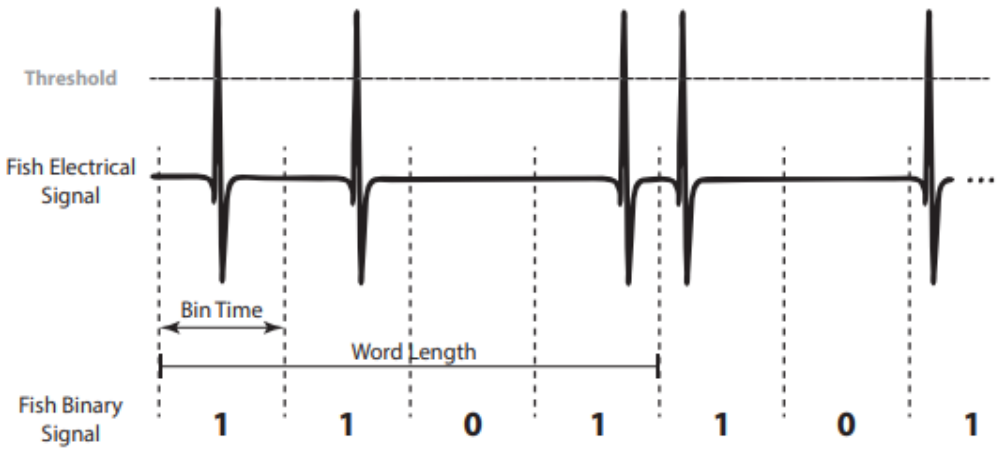

Figura 2 Codificación de un código binario de 4 bits. Figura extraída de [1].

<

Además, los códigos se detectan con una superposición, es decir, se toma un tamaño correspondiente al número de bits que van a formar las palabras detectando en un inicio la palabra i-ésima de tal forma que, para comprobar el resto de las palabras, se realizarán desplazamientos bit a bit tal y como se ve en la Figura 3.

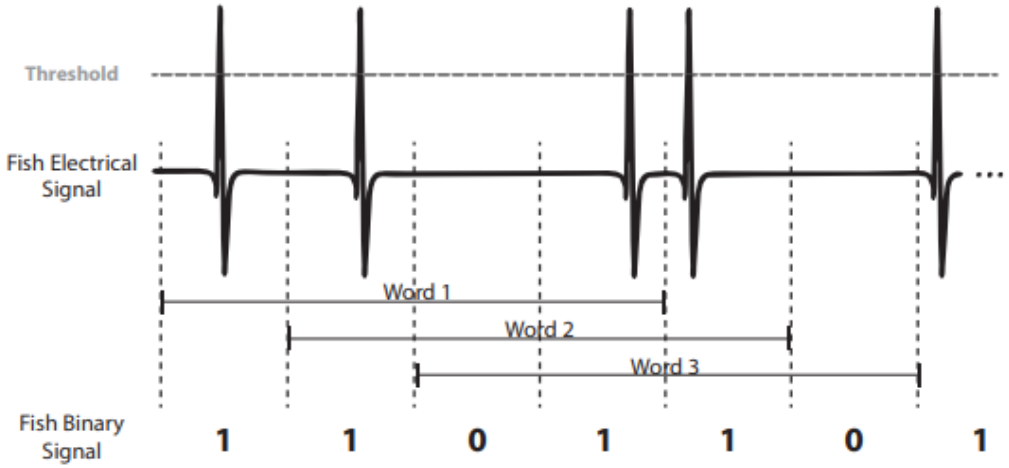

Figura 3 Codificación de varios códigos de 4 bits con una superposición de 3 bits. Figura extraída de [1].

4.3 Protocolo de estimulación (TCDS)

Como se ha mencionado anteriormente, uno de los principales objetivos de este trabajo es el desarrollo y la implementación del protocolo de estimulación TCDS [1, 2] que define cómo se establecerá la comunicación bidireccional con el sistema biológico.

Este protocolo de estimulación en ciclo cerrado ha sido validado en el contexto de la electrorrecepción en peces eléctricos.

La base de este protocolo de estimulación es la codificación temporal de la señal en palabras binarias, es decir, adquirimos el voltaje en tiempo real de la señal emitida por el pez y se digitaliza de forma binaria en función de si recibimos un pulso o no en un tiempo determinado por RTXI tal y como se explica en la sección anterior, se procesa la señal y se envían estímulos en función de la entrada que recibe el sistema artificial.

Tal y como se especifica en la Figura 4 el protocolo de estimulación diseñado para este proyecto funciona de la siguiente forma. Para cada periodo establecido en el panel de control de RTXI, se adquiere el voltaje de la señal emitida por el pez en tiempo real y a continuación, se codifica de forma binaria la señal. Cuando se detecta la codificación de una palabra con el mismo tamaño que la palabra que se está buscando se comparan, si coinciden, al final del bin comienza la estimulación.

Cabe destacar que el algoritmo diseñado funciona en ventanas de tiempo, durante estos periodos de tiempo, 60 milisegundos por ejemplo (con una frecuencia de muestreo de la señal de 10 kHz), se capta el voltaje de la señal emitida por el pez en intervalos de tiempo más pequeños (establecidos en el periodo del sistema) con el objetivo de “dibujar” de la forma más precisa la señal que se está detectando. Cuando se detecta un pulso por encima de un umbral determinado a esa ventana de tiempo se le asigna un 1, es decir, este será el

siguiente bit que formará la palabra binaria. A continuación, se comprueba si es la palabra que se está detectando y se estimula en función de ello. Por el contrario, si se agota el tiempo establecido para esta ventana y no se ha detectado ningún pulso por encima de dicho umbral el siguiente bit que formará la palabra será un 0, se comprueba si es la palabra buscada y se estimula en función de esta comparación. Es importante saber que la estimulación tiene lugar siempre en el mismo instante, es decir, al finalizar la ventana de tiempo o el tiempo de bin, aunque la detección de la palabra se haga en momentos diferentes, esto es lo que hace de este protocolo algo básico y estándar, ya que se puede modificar de muchas maneras igualmente válidas y a estudiar a futuro como la estimulación aplicando un retraso fijo, variable, etc.

Por último, al sobrepasar el número de bits de los que puede estar formada una palabra, al detectar el siguiente bit se realiza un desplazamiento, dando lugar a una nueva palabra, tal y como se ha explicado anteriormente.

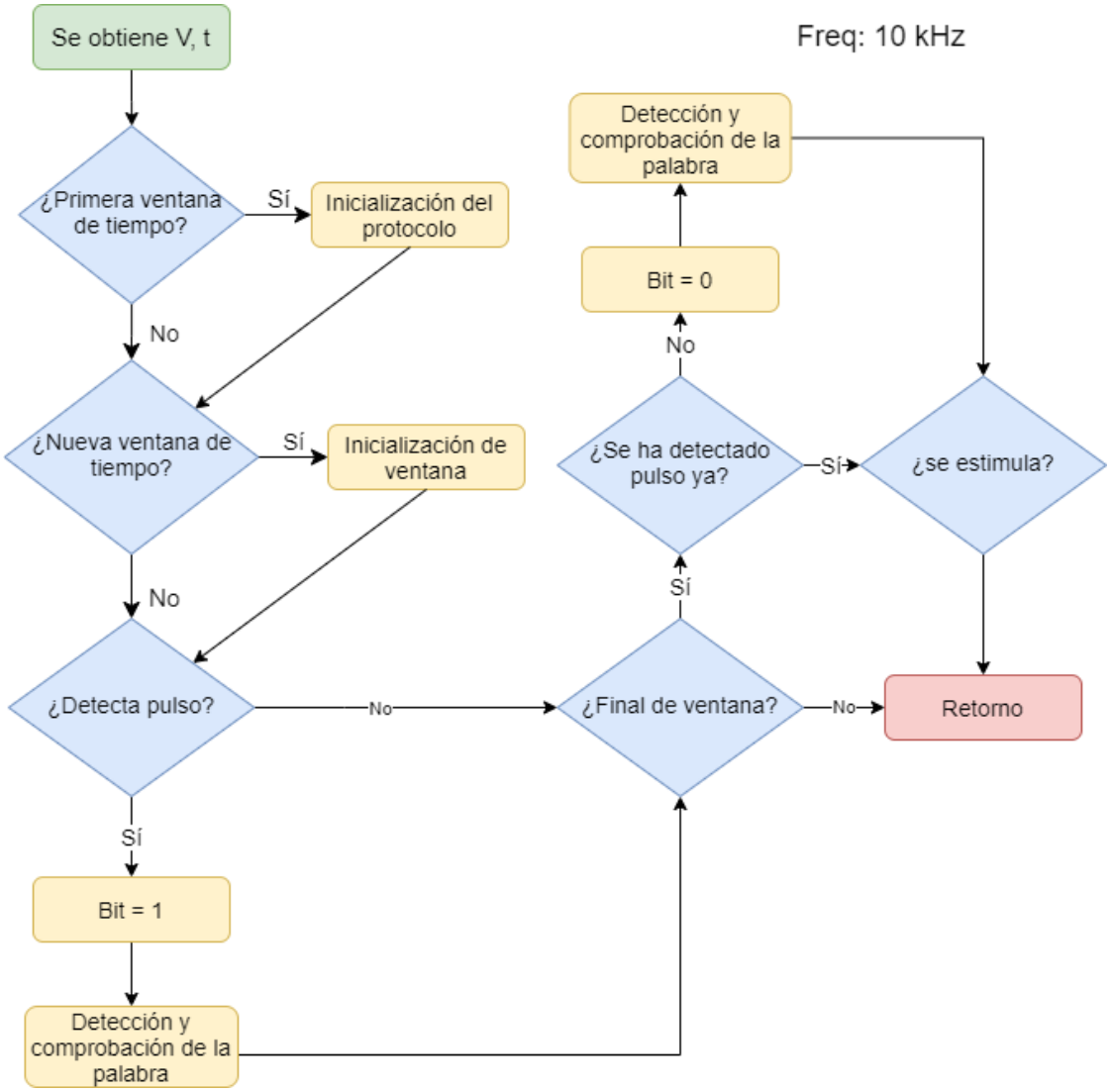

Figura 4 Diagrama de flujo de control del protocolo de estimulación en ciclo cerrado (TCDS). Cabe destacar que este protocolo se realiza tantas veces por segundo como se indique en la frecuencia de muestreo especificada en el experimento. Por ejemplo, si se ha especificado una frecuencia de muestreo de 10 kHz, este protocolo se realiza 10 e4 veces por segundo con la precisión requerida.

5 Análisis y diseño de la aplicación

5.1 Análisis de requisitos.

Los requisitos no funcionales de la aplicación desarrollada en este proyecto son:

- La herramienta deberá ser capaz de trabajar con latencias bajas.
- En tiempo real la actualización de parámetros debe darse bajo condiciones estrictamente marcadas y de forma constante.

Los requisitos funcionales desarrollados en este proyecto son se pueden observar en la Figura 5.

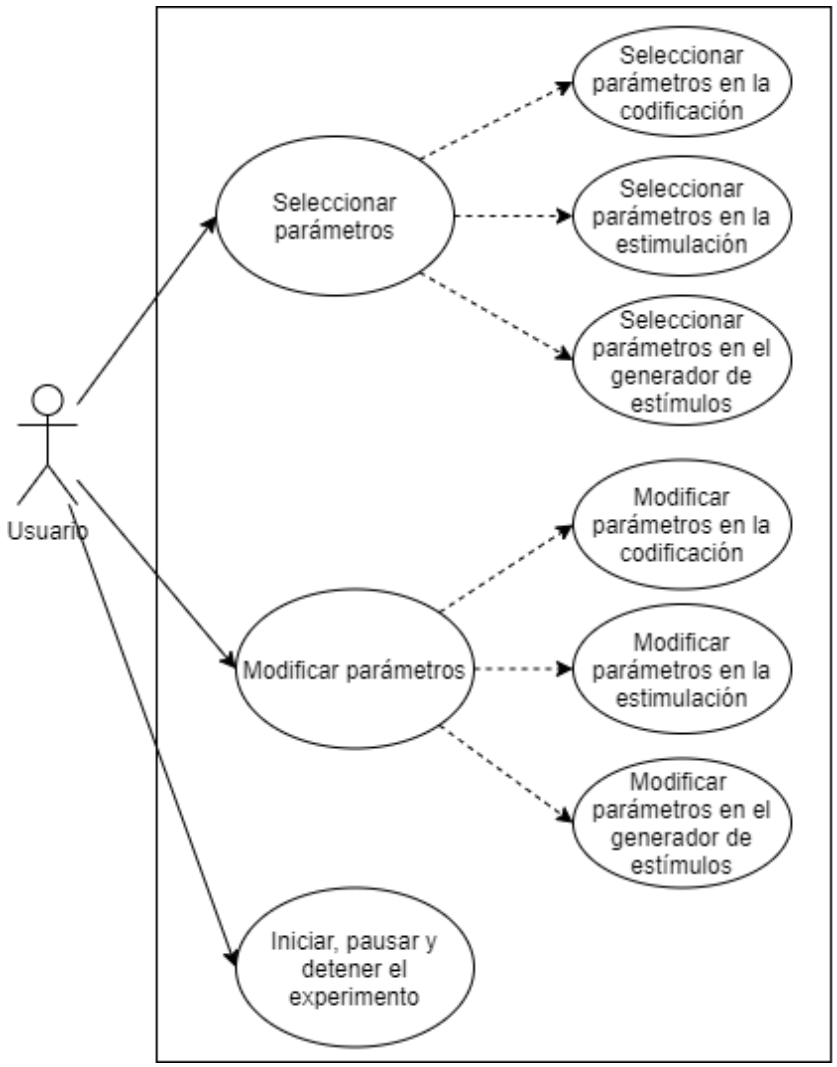

Figura 5 Diagrama de casos de uso.

- Selección de parámetros en la codificación: El usuario debe poder variar los siguientes parámetros:
  - ❖ Tiempo de duración del bin.
  - ❖ Tamaño de la palabra.
  - ❖ Palabra para detectar.

- ❖ Voltaje umbral.
- ❖ Máximo de palabras.

- Selección de parámetros en el generador de estímulos: El usuario debe poder variar los siguientes parámetros:
  - ❖ Fichero de carga.
  - ❖ Tamaño del buffer de datos.
- Selección de parámetros en la estimulación: El usuario debe ser capaz de variar los siguientes parámetros:
  - ❖ Fichero de carga.
  - ❖ Número de datos del estímulo.
- Modificación de parámetros: El usuario debe ser capaz de cambiar los parámetros a su antojo tanto si el sistema está corriendo en tiempo real o como si no.
- Iniciar, pausar y detener: El usuario debe poder iniciar, pausar y detener la ejecución en tiempo real del experimento.

5.2 Diseño de la herramienta

Como se ha mencionado anteriormente, una de las principales ventajas del uso de RTXI es la modularidad. Esto permite dividir la estructura de este proyecto en diversos módulos y cada uno de ellos diseñado para realizar una función específica.

De esta forma, la estructura de este proyecto estará constituida por módulos que se conectan entre sí por sus entradas y salidas formando una arquitectura funcional tal y como se ve en la Figura 6. Estos módulos serán capaces de generar información, tratarla y generar eventos en función de los datos obtenidos.

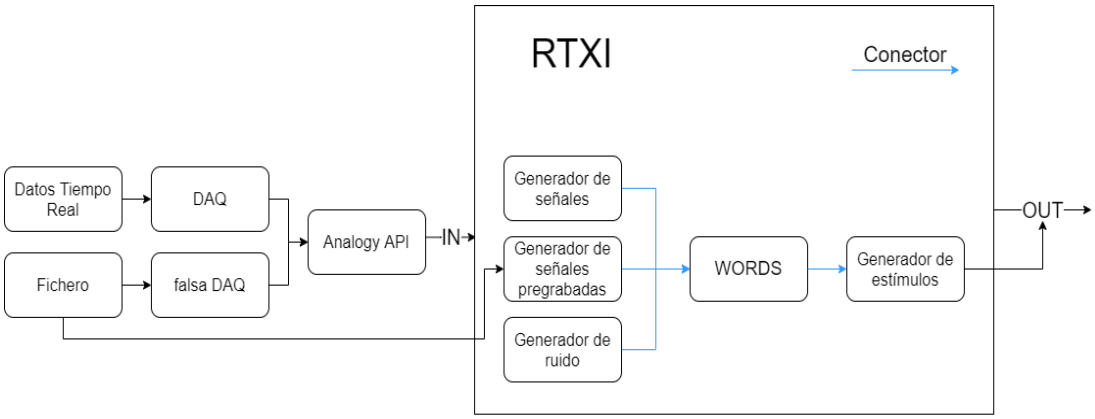

Figura 6 Arquitectura del proyecto.

Para generar datos y probar el funcionamiento del protocolo implementado en este proyecto se usan tres módulos, el ***Generador de ruido*** y el ***Generador de señal***, módulos propios del sistema de RTXI y el ***Generador de señales pregrabadas*** desarrollado para simular la señal del pez como registraría los datos la tarjeta de adquisición de datos del laboratorio, ya que fue imposible ir a éste por el reciente confinamiento vivido. El módulo ***WORDS*** será el que se encargue de tratar la información recibida en tiempo real y decidirá si se estimula o no al pez también en tiempo real. Por último, el ***Generador de estímulos*** será el encargado de precargar los estímulos y lanzarlos en tiempo real cuando

el módulo *WORDS* se lo indique. También disponemos del módulo ***Osciloscopio*** que ayuda con el depurado de la herramienta, y el módulo ***Conector*** que se encargará de unir las entradas y salidas de los módulos en tiempo real, ambos propios de RTXI. Los módulos unidos por el Conector son aquellos que están relacionados mediante una flecha de color azul en la Figura 6.

5.2.1 Generador de ruido

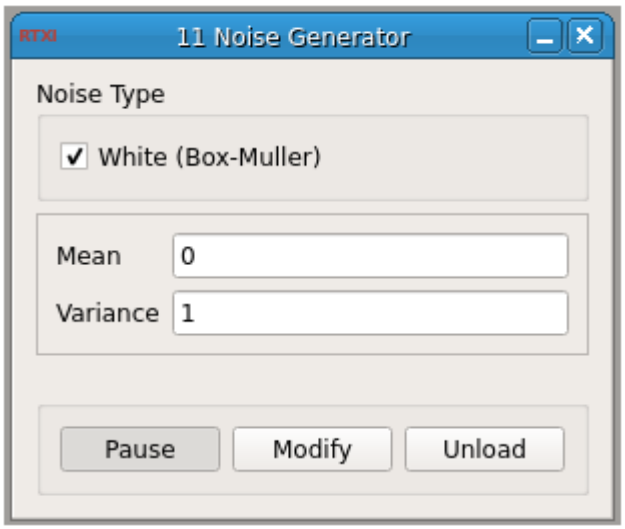

Figura 7 Interfaz del módulo generador de ruido. En este módulo se puede seleccionar la media y la varianza con la que se generarán los datos de la señal de ruido.

Con este módulo propio de RTXI, se puede generar ruido a partir de los parámetros que tiene de entrada, la media y la varianza. El ruido generado será la entrada de voltaje que recibirá el módulo *WORDS*. Como se puede ver en la Figura 7, a este módulo se le puede especificar la media y la varianza de la señal que generará.

5.2.2 Generador de señal

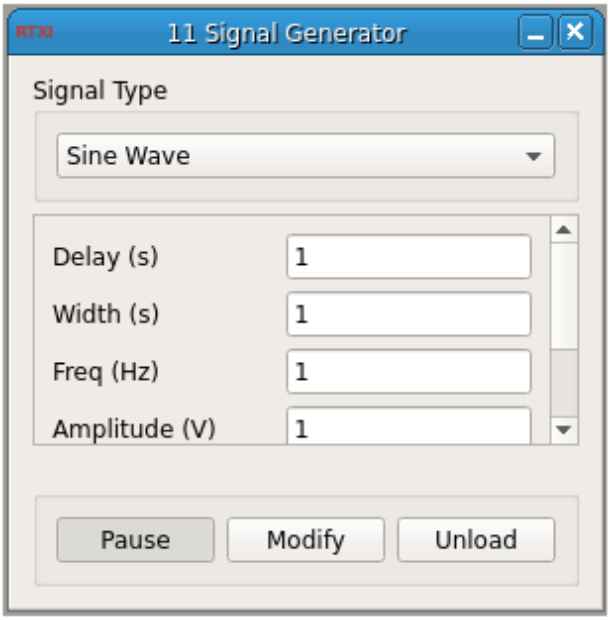

Figura 8 Interfaz del módulo generador de señales. En este módulo se pueden seleccionar diferentes parámetros en función de la señal a generar, ya sea sinusoidal, cuadrada, de sierra o ZAP.

Como se puede ver en la Figura 8, con este módulo, propio de RTXI, podemos crear las siguientes señales:

- Onda sinusoidal: necesita los parámetros de frecuencia y amplitud.
- Onda cuadrada monofásica: necesita retraso, ancho y amplitud.
- Onda cuadrada bifásica: necesita retraso, ancho y amplitud.
- Onda diente de sierra: necesita retraso, ancho y amplitud máxima.
- ZAP (*Impedance Amplitude Profile*): necesita frecuencia inicial y final, amplitud y duración.

Al igual que en los demás generadores, también se conectará la salida resultante a la entrada del módulo *WORDS* en tiempo real.

5.2.3 Generador de señales pregrabadas

Este módulo desarrollado en este proyecto tiene como objetivo cubrir una necesidad que surge durante el desarrollo del proyecto. Debido al confinamiento que ha tenido lugar era imposible hacer uso del laboratorio para el desarrollo del proyecto, es por esto por lo que se ha implementado este módulo para la simulación de los impulsos eléctricos producidos por la señal emitida por el pez. Este módulo precarga un estímulo leyendo datos de un fichero específico y los transmite en tiempo real simulando el comportamiento del sistema biológico al igual que se hace en las pruebas del laboratorio.

Este módulo no tiene ninguna entrada ya que es el encargado de generarla, como salida tiene la señal que simula y como parámetros el usuario puede especificar el fichero que quiere cargar y la cantidad de datos que formarán la señal.

5.2.4 WORDS

Este módulo desarrollado en este proyecto es el encargado de analizar y monitorizar los datos recibidos como señal del pez eléctrico con el objetivo de garantizar que se lleve a cabo el protocolo de estimulación diseñado específicamente para este proyecto. Es decir, es el encargado de analizar el voltaje producido por el campo eléctrico y enviarle un mensaje al módulo generador de estímulos indicándole que tiene que iniciar el proceso de estimulación.

Como entrada a este módulo llegará el voltaje producido por el pez o por cualquiera de los generadores de señal que forman parte de la estructura del proyecto. Además, el usuario puede modificar el tamaño de la palabra, la palabra que el algoritmo debe detectar, el umbral de voltaje para detectar un pico y el tamaño del bin a través de los parámetros visibles en la interfaz gráfica del módulo.

5.2.5 Generador de estímulos

Este módulo desarrollado en este proyecto es el encargado de emitir señales pregrabadas con las que se estimulará al sistema biológico con el objetivo de realizar un acto comunicativo bidireccional.

Como entrada tiene la salida del módulo *WORDS*, el cual se encargará de comunicarle en qué momento deberá comenzar a emitir el estímulo, como parámetros el usuario puede especificar el fichero que quiere cargar y la cantidad de datos que formarán la señal.

5.3 Decisiones de diseño

Durante el diseño y desarrollo de este proyecto se han tenido que llevar a cabo numerosas decisiones en cuanto a la estructura y funcionalidad de éste. Las decisiones más importantes que afectan tanto a la estructura del proyecto como a los diseños de los posibles experimentos futuros son las siguientes:

Existe la posibilidad de que en un mismo bin se detecte más de un pulso o *spike*, por lo que había que decidir qué hacer si se diese este caso. En este tipo de situaciones, una vez el algoritmo ha detectado un pulso en una ventana de tiempo, el bit que codifica este bin pasa a valer uno y no se vuelve a detectar nada durante esta ventana de tiempo. Esta decisión puede afectar a la hora de estudiar el comportamiento del pez por lo que diversas soluciones se evaluarán como potencial trabajo futuro.

En RTXI existe un módulo llamado ***spike detector***, capaz de detectar pulsos en tiempo real por lo que es una funcionalidad muy útil que podría ahorrar esfuerzo y tiempo al trabajo. Sin embargo, no se opta por el uso de ese módulo debido a que es necesario realizar comprobaciones específicas a la vez que se detectan pulsos en tiempo real. Por ello, esta funcionalidad está incorporada al módulo *WORDS*.

Por circunstancias especiales (periodo de cuarentena debido a la pandemia de COVID19) no ha sido posible probar los protocolos de manera experimental con un pez eléctrico. Debido a estas circunstancias se incorporó al proyecto la funcionalidad de simular la señal del pez mediante la implementación de un generador de ondas pregrabadas.

También se valoró la posibilidad de desarrollar e instalar un falso controlador a bajo nivel que simulara la obtención de datos en tiempo real siguiendo la interfaz de Analogy. Sin embargo, tras su estudio, se optó por un generador como módulo de RTXI que aportaba la misma funcionalidad mediante un desarrollo más simple.

Como se ha mencionado anteriormente, RTXI es una herramienta que hace uso del tiempo real. Sin embargo, para poder usarla hay que saber cuál es su nivel de eficiencia a la hora de realizar las tareas que se van a desarrollar. Para comprobar que es eficiente hay que medir las latencias para las que se ejecutan las tareas en tiempo real. RTXI cuenta con un módulo que mide las latencias online pero no almacena los datos reflejados y únicamente son mostradas por pantalla. Por lo tanto, se ha desarrollado esta funcionalidad dentro del módulo *WORDS* almacenando estos resultados.

Otra de las decisiones importantes que se han tomado durante el diseño y el desarrollo de este trabajo es cómo debe comportarse el protocolo cuando ya se encuentra estimulando y detecta una palabra de las que disparan la estimulación. En este caso, se ha decidido no comenzar con un nuevo estímulo si la estimulación ya se está llevando a cabo. Esta decisión afecta al diseño del protocolo de estimulación y, por lo tanto, tiene una influencia considerable en el desarrollo de los experimentos.

6 Desarrollo

Como se ha mencionado anteriormente, el principal desarrollo que se ha llevado a cabo en este proyecto está constituido por los módulos generador de señales pregrabadas, *WORDS* y el generador de estímulos. La implementación de estos módulos se ha realizado en el lenguaje de programación C++.

El código de estos módulos se puede encontrar en el github del GNB (https://github.com/GNB-UAM/tcds-rtxi).

6.1 Generador de señales pregrabadas

Como se ha mencionado en el diseño de este proyecto (ver sección 5) este módulo simulará la señal emitida por un pez eléctrico en tiempo real, por lo que se le podrá dar a la señal del pez una forma específica. Esto supone una gran cantidad de ventajas. Como ya se ha comentado, este proyecto no iba a contar con este módulo ya que iba a aplicarse a la señal real adquirida en tiempo real de un sistema biológico vivo. Sin embargo, debido a la cuarentena como consecuencia de la pandemia de COVID19, se exploraron nuevas vías para el desarrollo de este proyecto que desembocaron en la implementación de este módulo.

El módulo generador de señales pregrabadas obtiene los datos de voltaje que formarán la señal emitida por el pez de un fichero y los envía en tiempo real al módulo *WORDS*, el cual, los tratará como si fuera la señal real emitida por el pez. De esta forma, podremos comprobar el correcto funcionamiento de los protocolos sin la necesidad de realizar un análisis con un sistema vivo. Este módulo ha sido una parte indispensable para el

proyecto al no poder hacer uso de los laboratorios debido a la cuarentena. Además, habilita la opción de testear nueva funcionalidad sin necesidad de contar con hardware de adquisición de datos y un sistema en tiempo real.

Para llevar a cabo su función este módulo consta de una cola específicamente diseñada para el envío de los datos en tiempo real al módulo *WORDS*, funcionando de la siguiente manera:

Al comienzo del experimento se instanciará este módulo dentro de RTXI, en este instante el módulo comienza con su funcionalidad, cargando una cantidad de datos finita especificada por los parámetros del módulo en una cola circular. Una vez comienza la ejecución del experimento, RTXI accederá a este módulo en tiempo real en los intervalos de tiempo definidos en el periodo de la herramienta, de esta forma el módulo enviará los datos precargados en tiempo real a una frecuencia determinada. Además, al utilizar una cola circular, enviará la misma cantidad de datos en bucle, asegurando el envío continuo de datos mientras el experimento se esté ejecutando (Figura 10).

Por último, cabe destacar que este módulo no recibe ninguna entrada ya que es éste el que la genera y tiene como salida el voltaje que envía al módulo *WORDS*. Como parámetros de la interfaz consta de un pequeño botón para elegir el fichero que usará para transmitir los datos y el número de datos que formarán la cola. Cada vez que se modifique alguno de estos parámetros se leerán de nuevo los datos y se volverá a construir la cola. En la Figura 9 se puede ver cómo es la interfaz gráfica de este módulo.

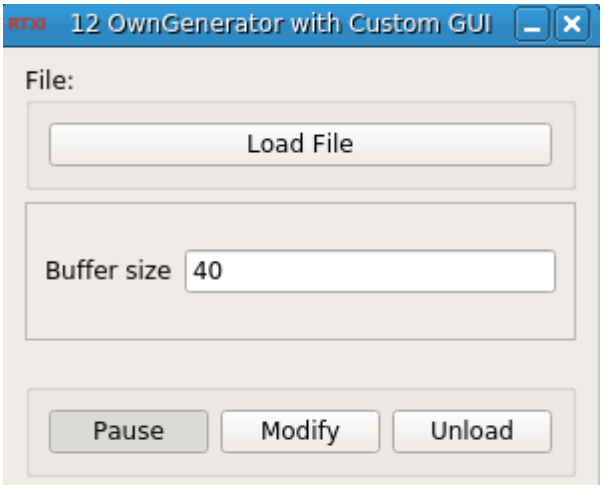

Figura 9 Interfaz del módulo generador de señales pregrabadas. En este módulo se puede seleccionar el fichero de datos que transmitirá y la cantidad de estos datos que se cargarán en el *buffer*.

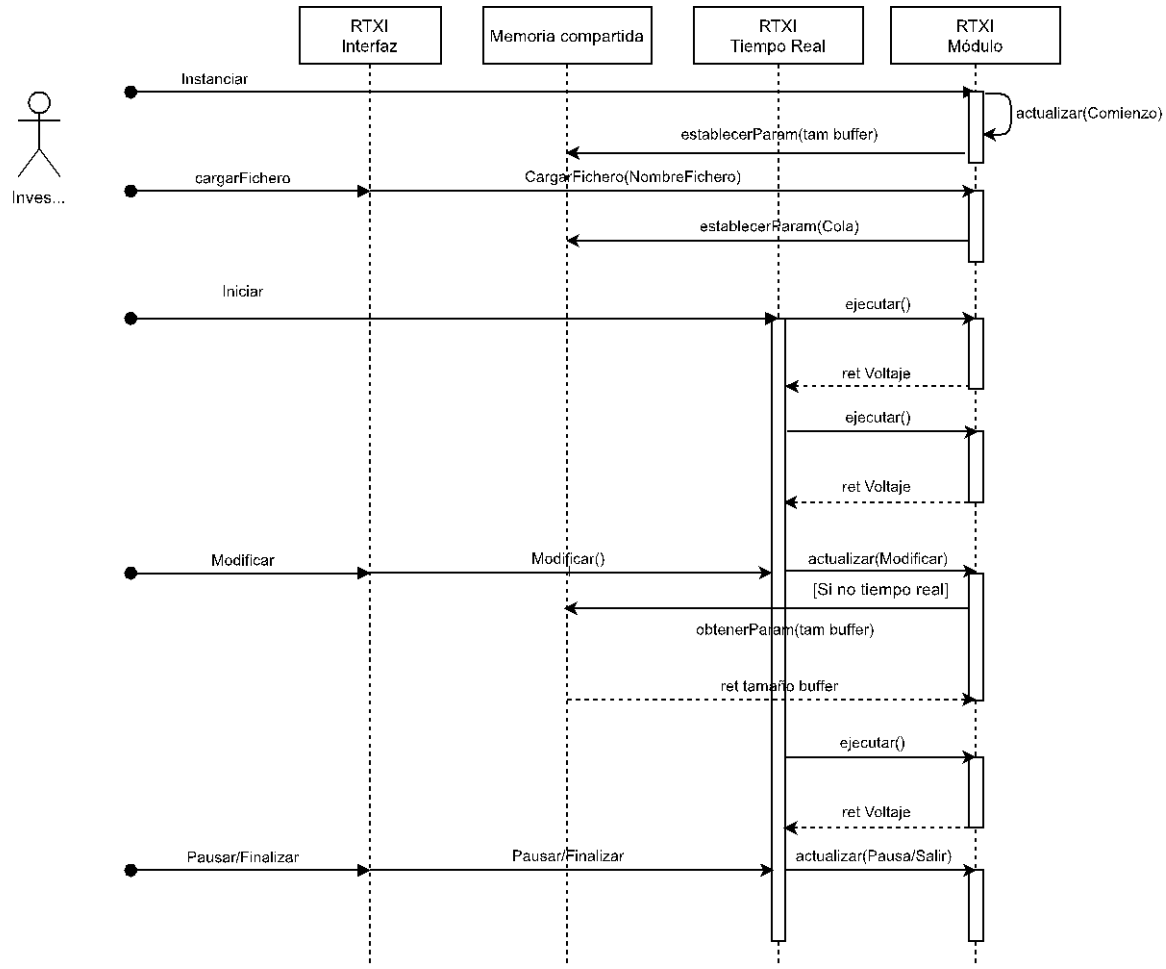

Figura 10 Diagrama de secuencia módulo generador de señales pregrabadas y el generador de estímulos.

En este módulo no hay ninguna entrada de la que obtener datos ya que es éste el encargado de recogerlos. Para empezar, es la parte del usuario la que se encarga de leer un estímulo y precargarlo, además, una vez iniciada la tarea existe una comunicación entre la interfaz y el tiempo real donde RTXI se encarga de comunicar ambas para la transferencia de información como bien puede ser el momento de iniciar la tarea, los parámetros especificados por el usuario, o el momento en el que se finaliza. Por último, la salida de este módulo y como ya se ha mencionado es la entrada del módulo *WORDS*.

6.2 WORDS

Aunque se podría decir que todos los módulos desarrollados en este proyecto son imprescindibles para poder configurar y realizar un experimento, este módulo desempeña el papel más importante y por lo tanto es el más crítico.

En este módulo es donde se implementa la digitalización binaria de la señal, el análisis de ésta mediante el uso del protocolo de estimulación diseñado para este proyecto y es el encargado de dar pie al módulo generador de estímulos para que empiece el proceso de estimulación.

El módulo *WORDS* está diseñado para ser el monitor de la estimulación en ciclo cerrado que está teniendo lugar en el experimento, ya que es el único módulo en el que se llevan a cabo distintas tareas de control y análisis de los datos.

Al instanciarse el módulo en el experimento se inicializan los parámetros modificables por el usuario a través de la interfaz gráfica diseñada para RTXI (Figura 11). El voltaje umbral a partir del cual se considera que se ha detectado un pulso, el tamaño de la palabra, es decir, el número de bits que la forman, la palabra que se va a detectar y el tiempo de duración de un bin. También se inicializan otro tipo de parámetros y variables que son utilizados para el análisis y el control del flujo de datos a través del protocolo de estimulación.

Una vez empieza la ejecución del protocolo en tiempo real el módulo obtiene el voltaje de la señal como entrada y registra el tiempo en el que este voltaje es leído, a continuación, detecta si ha tenido lugar un pulso dentro del bin en el que se están recibiendo los datos, si es así, añade un 1 al buffer de bits que forman la palabra y detecta si la palabra almacenada en ese bin es la palabra buscada, en caso de que no se detecte pulso comprueba si ha acabado el tiempo del bin. El algoritmo solo añade un cero a los bits que forman la palabra si ha terminado el tiempo del bin y no se ha detectado ningún pulso, además, se comprueba de nuevo si se trata de la palabra que el algoritmo está buscando. En ambos casos, se detecte pulso o no, solo se estimulará si se ha detectado la palabra siempre al final del bin (Figura 12).

Todo este proceso se realiza cada vez que tiene lugar un periodo en tiempo real, es decir, si por ejemplo la frecuencia de muestreo es de 20 kHz, el protocolo se realiza 20.000 veces por segundo con la precisión requerida. Además, este módulo consta de una salida que sirve como bandera para activar el proceso de estimulación, es decir, cada vez que se ejecuta este módulo, la salida será un 0 siempre y cuando no se haya detectado la palabra y no se tenga que llevar a cabo la estimulación, sin embargo, en el momento que se detecte la palabra, el módulo enviará como salida un 1 indicando que se tiene que estimular al sistema biológico.

La salida de este módulo la recibe el generador de estímulos como entrada, quien está en modo “espera” hasta que recibe la señal que le permite iniciar el proceso de estimulación.

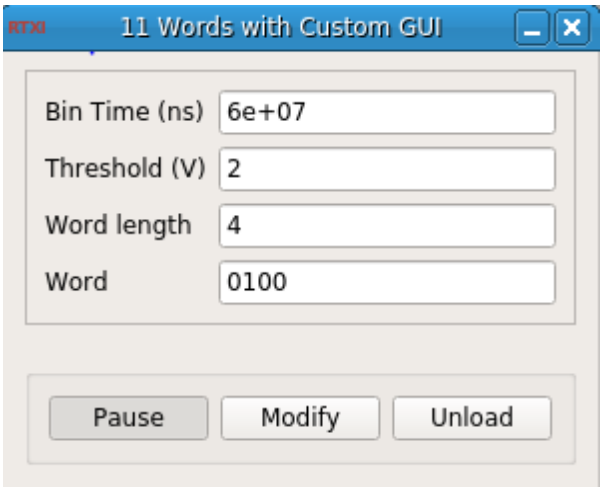

Figura 11 Interfaz del módulo *Words*. En esta imagen se puede ver un ejemplo de ejecución utilizado para probar el módulo en el laboratorio, con un tamaño de palabra de 4 bits, un máximo de 5 palabras, el número 4 como palabra a buscar, un umbral de 2V para detectar pulso y 60 ms de tiempo de bin.

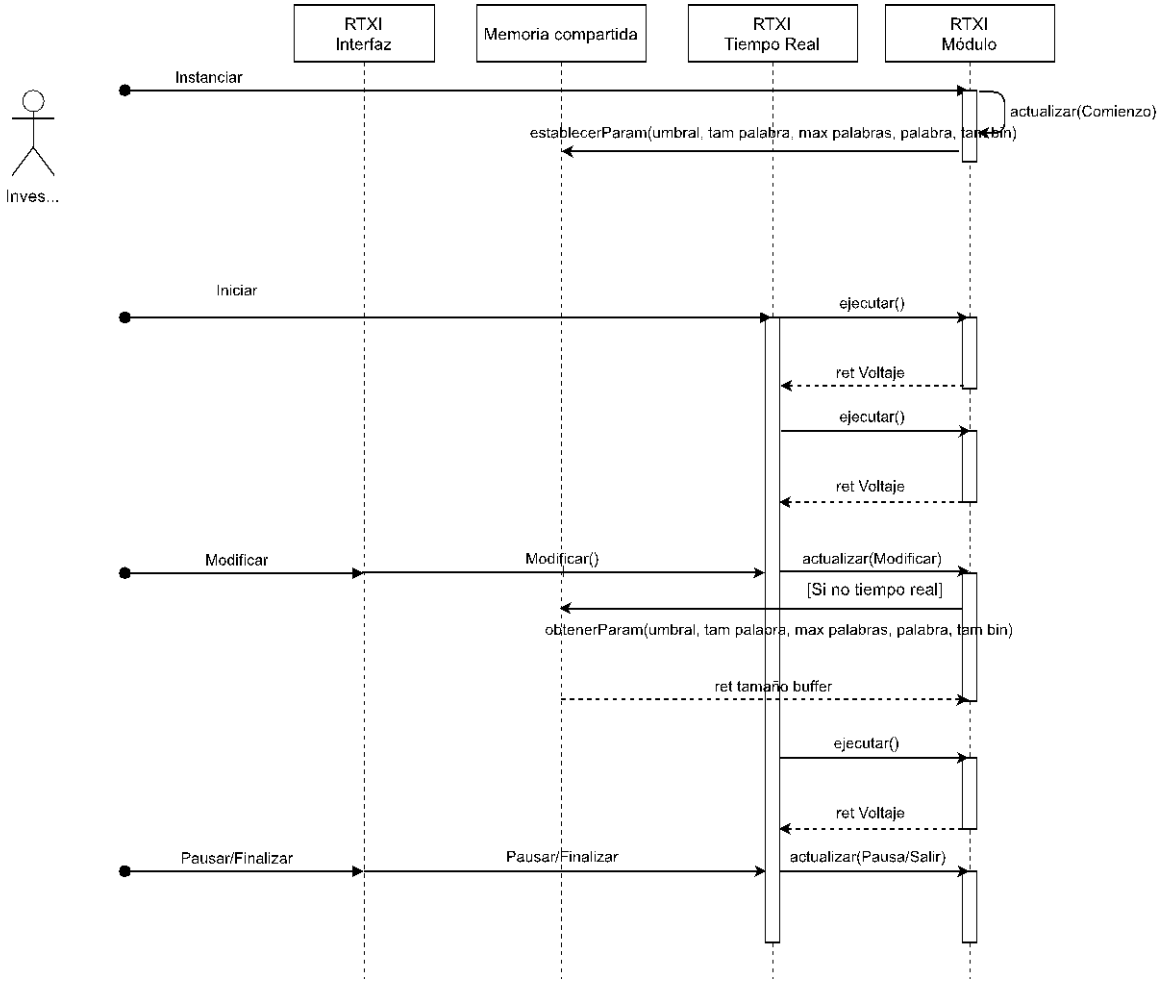

Figura 12 Diagrama de secuencia módulo *WORDS*.

6.3 Generador de estímulos

La funcionalidad desarrollada en este módulo es clave para establecer una comunicación bidireccional con el sistema biológico, es decir, es imprescindible para el desarrollo de un protocolo de estimulación basado en ciclo cerrado.

A nivel estructural y funcional es bastante parecido al módulo generador de señales pregrabadas (Figura 13). Al igual que éste, el generador de estímulos cargará un conjunto de datos leído de un fichero al instanciar el módulo configurando el experimento en RTXI, una vez comience el experimento y cuando sea oportuno, el módulo transmitirá los datos precargados como salida.

Sin embargo, existen algunas diferencias respecto al módulo generador de señales pregrabadas. Este módulo no dispone ni hace uso de una cola circular, es decir, se cargará un estímulo y cuando sea necesario se transmitirá en tiempo real por su salida, de tal forma que cuando envíe el último dato terminará de enviar el estímulo.

Además, este módulo cuenta con una entrada, a diferencia de generador de señales pregrabadas. Esto se debe a que al instanciar el módulo se precargará el estímulo, quedando en un estado latente esperando a recibir una señal adaptando la funcionalidad de un disparador. A cada periodo especificado en RTXI y durante la ejecución en tiempo real este módulo lee la entrada que recibe, manteniéndose a la espera de recibir un 1 como señal de entrada, momento en el que comenzará a emitir el voltaje que forma la señal previamente cargada. El diagrama de secuencia de este módulo es igual que el del módulo generador de señales pregrabadas que se muestra en la Figura 10.

Es importante aclarar que mientras este módulo esté emitiendo un estímulo, ignorará cualquier señal que reciba, de tal forma que no se podrá volver a estimular hasta que termine de enviar la señal precargada. Por último, los parámetros de los que consta este módulo son los mismos que tiene el generador de señales pregrabadas.

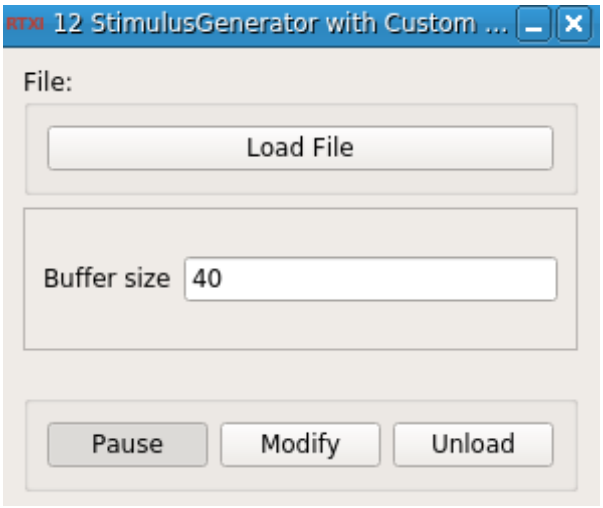

Figura 13 Interfaz del módulo generador de estímulos. En este módulo se puede seleccionar el archivo de datos que precargará como estímulo, así como el número de datos que formarán el buffer que se transmitirá.

7 Integración, pruebas y resultados

7.1 Comprobación del correcto funcionamiento del protocolo

El primero de los experimentos realizados en el laboratorio consistió en realizar una prueba cuyo propósito es saber si podemos usar RTXI en un laboratorio y en experimentos donde el factor físico interviene, es decir, podemos usar una tarjeta de adquisición de datos con esta herramienta.

Para ello, simplemente se usa el módulo generador de señal para transmitir una onda sinusoidal y se conecta la salida de este módulo a la salida de la DAQ, configurada como **analogy0** en RTXI. Además, se conecta la salida de la DAQ a un osciloscopio del laboratorio y a uno de los canales de entrada de la misma tarjeta, a la vez que mostramos por el módulo osciloscopio de RTXI la salida y la entrada de la tarjeta. De esta forma y tal y como se puede ver en la Figura 14 y en la Figura 15, se comprueba que es viable el uso de RTXI con los elementos hardware que hay en el laboratorio viendo cómo se genera la onda en el osciloscopio del laboratorio y pintando la entrada y la salida de la tarjeta por el osciloscopio de RTXI.

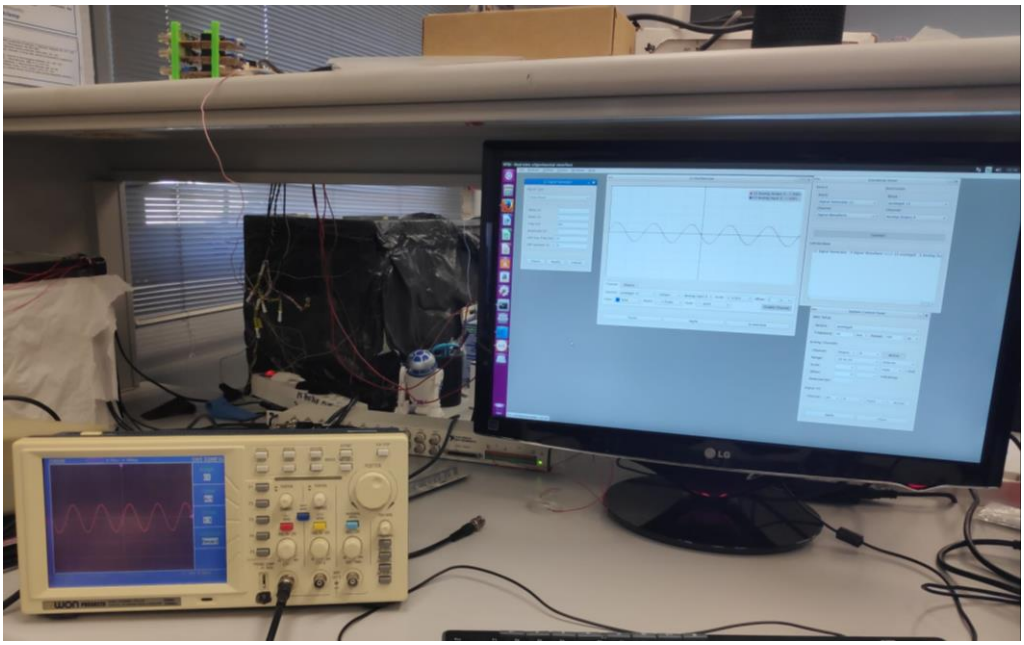

Figura 14 Montaje de RTXI básico junto con el hardware del laboratorio.

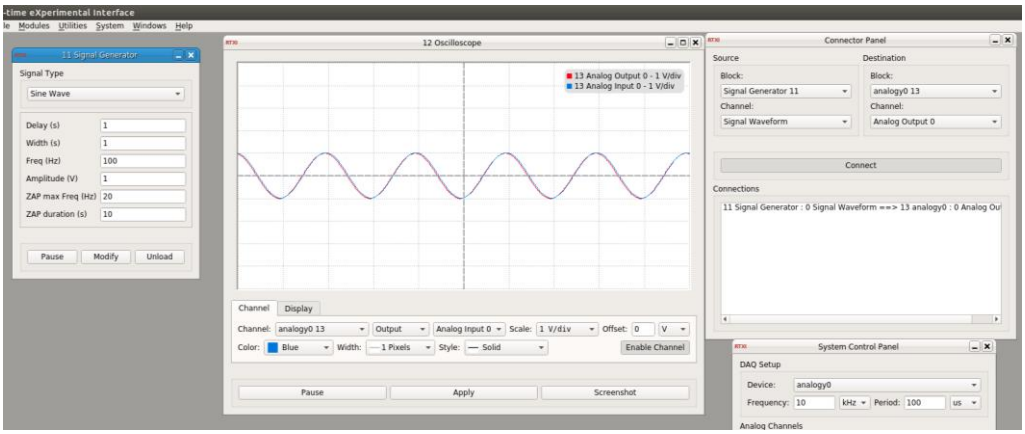

Figura 15 Composición del primer experimento en la interfaz de RTXI.

Tal y como se puede ver en la Figura 16, se observa cómo se dibuja la onda para ambos canales con un poco de desfase, esto significa que hay algo de latencia entre ellos, sin embargo, casi no se aprecia, otra señal de cómo RTXI trabaja con bajas latencias para frecuencias de 10 kHz que es la frecuencia con la que se realizan todos los experimentos.

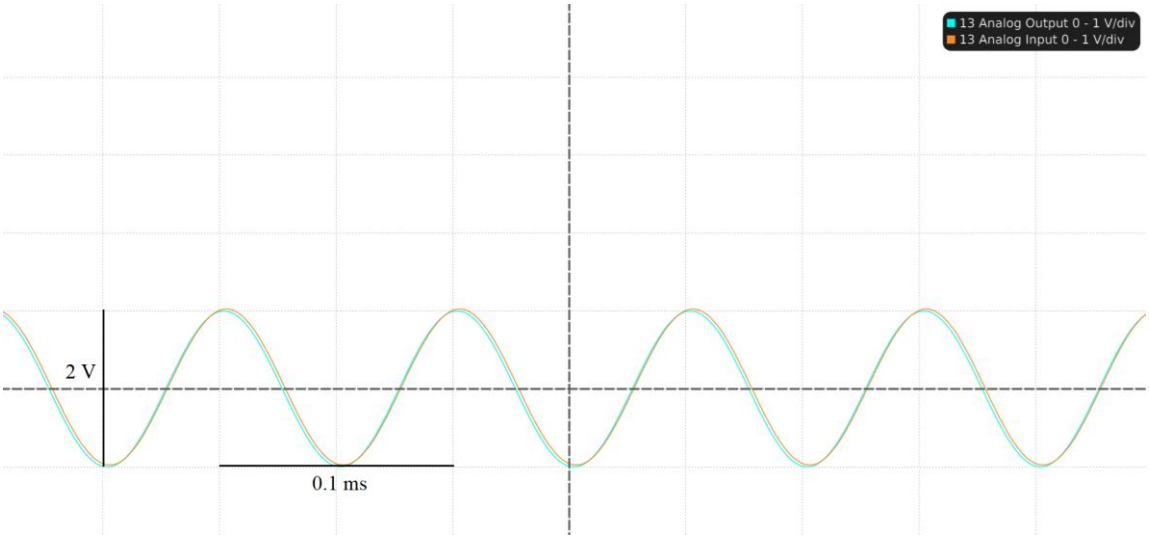

Figura 16 Representación del voltaje de entrada y de salida de la tarjeta de adquisición de datos en el osciloscopio de RTXI.

A continuación, se prueba el correcto funcionamiento de uno de los módulos desarrollados para este proyecto, el generador de señales precargadas, para ello, simplemente se realizó el experimento anterior cargando un pulso de un fichero grabado de la señal emitida de un pez eléctrico.

En la Figura 17 se puede ver cómo se ha trabajado en el laboratorio para probar el correcto funcionamiento de este módulo, haciendo también uso del osciloscopio del laboratorio.

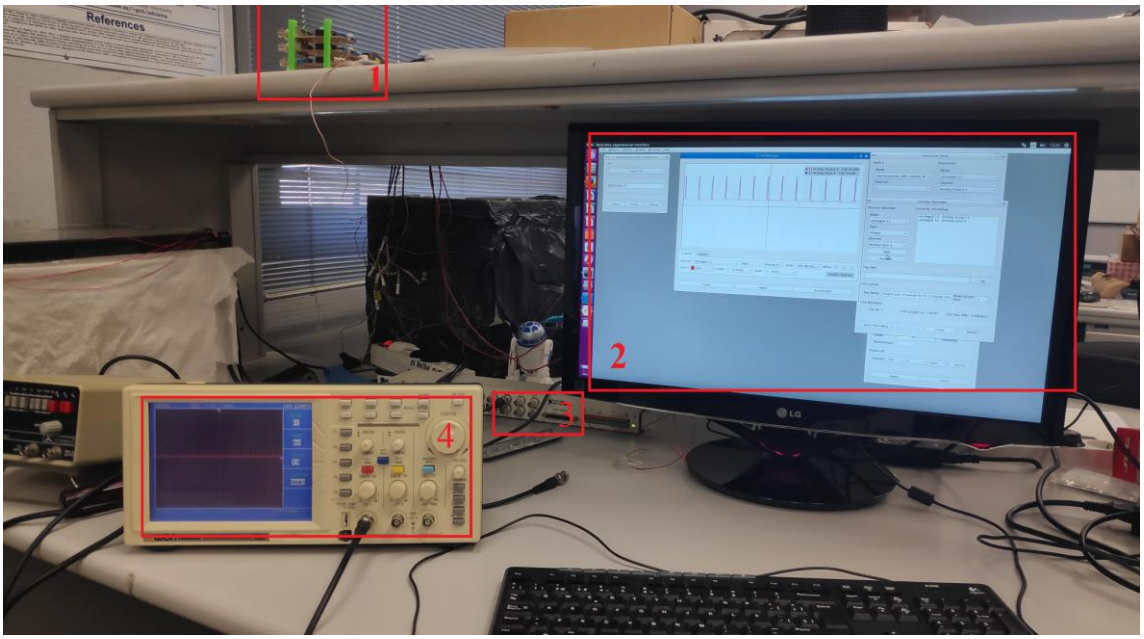

Figura 17 Montaje en el laboratorio para probar el correcto funcionamiento del módulo generador de señales pregrabadas. El número 1 de la imagen muestra el circuito sumador, el 2 la interfaz del sistema de tiempo real, el 3 la tarjeta de adquisición de datos y el 4 el osciloscopio del laboratorio.

Como se puede ver en la Figura 18 y la Figura 19, el osciloscopio recoge con una latencia mínima tanto el canal de entrada como el de salida de la tarjeta de adquisición de datos y ambos dibujan el mismo pulso.

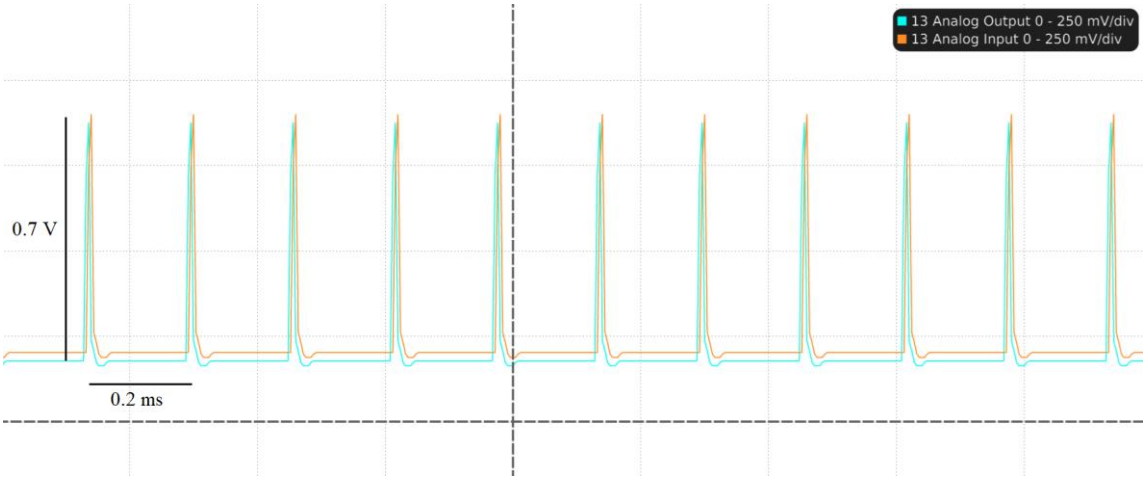

Figura 18 Representación en el osciloscopio de RTXI la entrada y la salida de la tarjeta de adquisición de datos con un estímulo corto cargado en el generado de señales pregrabadas.

Para terminar con esta fase de los experimentos se cargaron dos minutos de un fichero de pulsos grabados de un pez eléctrico comprobando que el módulo era capaz de cargar un estímulo de un tamaño considerable mostrando el voltaje de los canales de la DAQ por el osciloscopio (Figura 19).

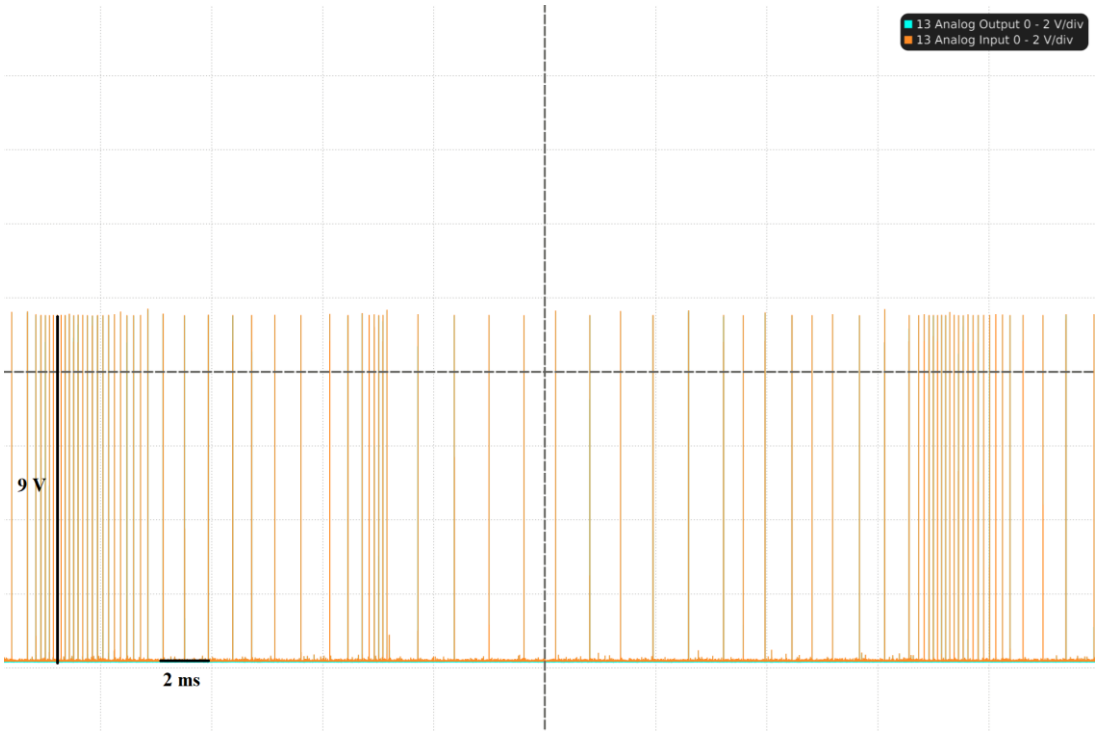

Figura 19 Representación en el osciloscopio de RTXI la entrada y la salida de la tarjeta de adquisición de datos con un estímulo de larga duración cargado en el generado de señales pregrabadas.

La siguiente fase de los experimentos consistió en simular la adquisición del voltaje emitido por un pez eléctrico artificial en el agua.

Como ya se ha mencionado, este trabajo fin de grado ha sido realizado durante un periodo de cuarentena por lo que ha sido inviable realizar experimentos con un pez eléctrico real, consecuencia de este suceso se desarrolló un módulo capaz de simular la emisión de campo eléctrico del pez, el generador de señales pregrabadas ya probado. Por lo que se sustituyó al pez eléctrico real por un dipolo emisor conectado a la salida de la tarjeta de adquisición de datos de forma que emite el mismo tipo de señal que la que emite un pez eléctrico real.

Como se ha mencionado, para este experimento se conectó la salida del generador de señales precargadas a la salida de la DAQ que a su vez se conectó al dipolo emisor que se sumergió dentro de una pecera. Un ejemplo de cómo se trabajó con este montaje se puede ver en la Figura 20.

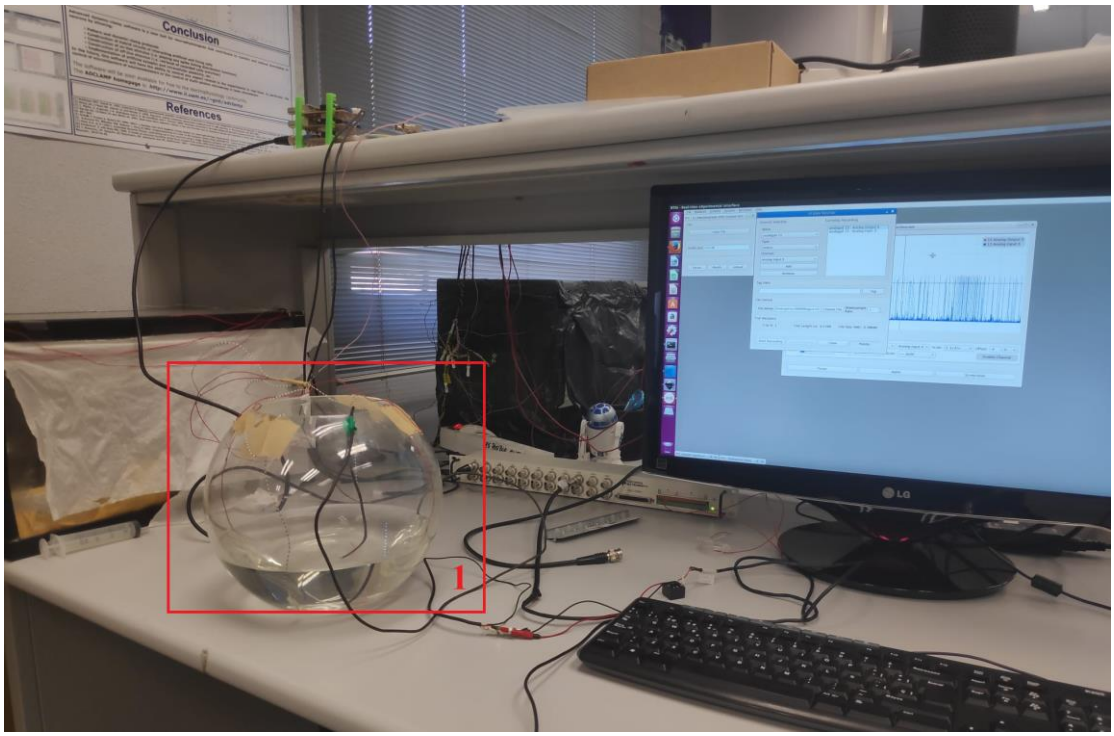

Figura 20 Montaje en el laboratorio haciendo uso de todo el hardware disponible para el experimento, la DAQ, la pecera, los dipolos y el circuito sumador. En el número 1 de la imagen se muestra la pecera con los dipolos.

Esta pecera también cuenta con dipolos receptores situados en los puntos cardinales, de tal forma que, al conectarlas a un circuito sumador desarrollado por el GNB, se realiza un proceso de sumado de señales dando lugar a una señal única [29, 1, 2].

La salida de este circuito sumador se conecta a la entrada de la DAQ, de esta manera se puede pintar en el osciloscopio de RTXI la entrada y la salida de la tarjeta de adquisición de datos.

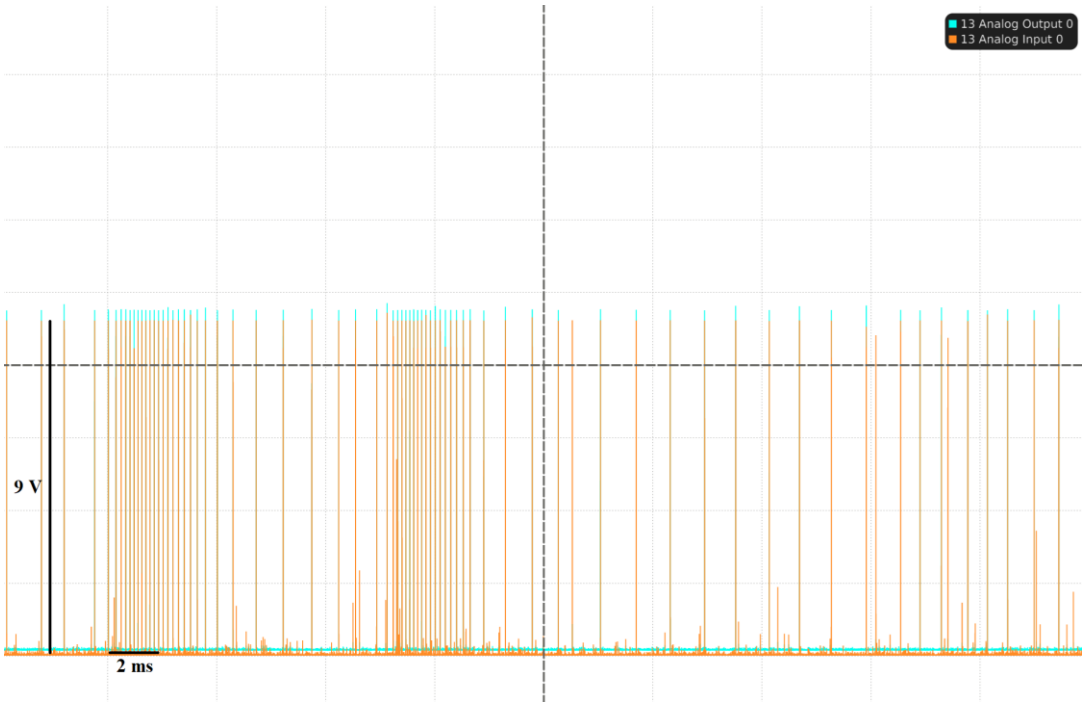

Figura 21 Representación en el osciloscopio de RTXI la entrada y la salida de la tarjeta de adquisición de datos simulando el pez eléctrico con el dipolo.

Como se puede apreciar en la Figura 21, el voltaje emitido por el módulo es el mismo que el recogido por la entrada de la tarjeta de adquisición de datos, por lo que se puede asumir que el módulo funciona correctamente y además podemos simular el envío de señales tal y como lo haría un pez eléctrico.

También se puede observar en la imagen que la señal recogida por la DAQ produce algo de ruido y pierde ganancia, esto es debido a que la señal producida por el generador está pasando por el agua y la pecera no está completamente aislada de todo lo que hay a su alrededor.

Por último, se comprobó cómo era el funcionamiento completo de todos los módulos desarrollados para este proyecto, es decir, del generador de señales pregrabadas, el módulo *WORDS* y el generador de estímulos. Para ello se utilizó el mismo montaje ya usado en la prueba anterior, sin embargo, ya no se comprobó que la entrada de la tarjeta de adquisición de datos obtuviera el mismo voltaje que el emitido por el generador de señales. En este nuevo experimento se conectó la entrada de la DAQ a la entrada al módulo *WORDS* el cual la analiza siguiendo el protocolo explicado en la metodología del trabajo. A su vez, la entrada del generador de estímulos estará conectada a la salida del módulo *WORDS*, esperando a que éste le informe si emitir o no el estímulo que tiene cargado.

Para esta comprobación se configuró el panel de control de RTXI para que las tareas en tiempo real se realizaran con una frecuencia de 10 kHz, el tamaño de las palabras buscadas por el módulo *WORDS* era de 4 bits, el tiempo de bin de 60 milisegundos y un voltaje umbral de 2 voltios para evitar el ruido.

Se cargó en el generador de señales un estímulo de casi dos minutos por lo que se grabó la señal de entrada al módulo *WORDS* y la salida del generador de estímulos con el módulo *Data Recorder* durante 110 segundos.

De esta manera se comprobó que siempre que el al módulo *WORDS* le entraba la cadena 0100 de bits, es decir, un 4 en decimal, el módulo generador de estímulos emitía el estímulo que tenía precargado tal y como se ve en la Figura 22.

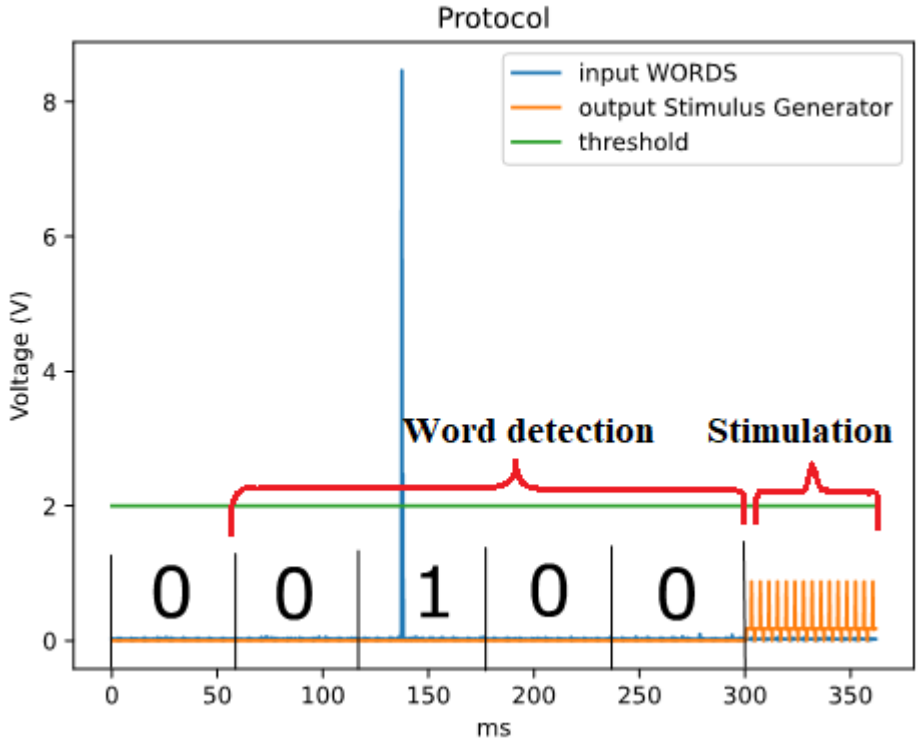

Figura 22 Ejemplo de comprobación del correcto funcionamiento del módulo *WORDS* simulando un pez eléctrico. El proceso de estimulación comienza en el siguiente periodo de tiempo a la detección de la palabra configurada. Este correcto funcionamiento del protocolo se cumple en todos los experimentos llevados a cabo en el laboratorio,

7.2 Pruebas de latencias

Una parte importante de este trabajo es comprobar que RTXI es una herramienta con la que se pueden realizar experimentos de ciclo cerrado en tiempo real en el contexto de los peces eléctricos.

Como ya se ha mencionado anteriormente, las acciones relacionadas con la comunicación bidireccional con el pez se realizan en el orden de los milisegundos, es por esto por lo que las tareas de los módulos en tiempo real deben ejecutarse con una frecuencia lo suficientemente alta para que los datos con los que se trabaja en los experimentos puedan ser analizados con la mayor exactitud posible.

Sin embargo, en los entornos de desarrollo basados en tiempo real existe una diferencia de tiempo entre cuando debería ejecutarse una tarea y cuando se ejecuta realmente, denominada como latencia tal y como se puede ver en la Figura 23.

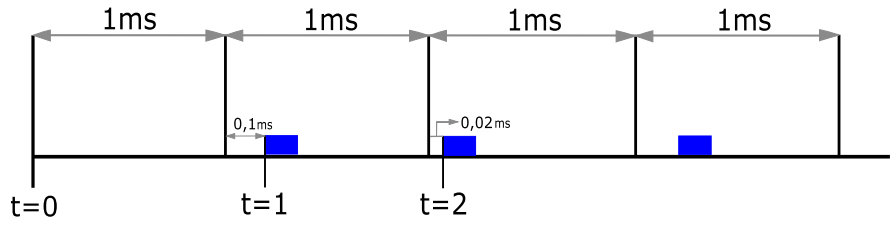

Figura 23 Esquema en el que se representa el concepto de latencia en un sistema en tiempo real. Los cuadrados azules son las tareas, y las líneas verticales indican cuando deberían realizarse, sin embargo, se ejecutan más tarde produciendo latencia.

Debido a este hecho, es muy importante saber que las tareas que se realizan en los módulos desarrollados en RTXI se ejecuten con la menor latencia posible.

Para comprobar que se puede usar RTXI para realizar este tipo de estudios se han realizado distintas pruebas de latencias en el laboratorio. En estas pruebas se utilizaban tanto los módulos desarrollados en este trabajo como los propios del sistema de RTXI junto con el hardware del laboratorio tal y como se especifica en el último experimento del apartado anterior. Es decir, se analizaron las latencias para el montaje completo del proyecto para distintas frecuencias, en concreto de 10 kHz, 15 kHz y 20 kHz, cubriendo un rango de valores más que suficiente en relación con la precisión que necesitamos para trabajar con estas señales biológicas.

Los resultados obtenidos fueron los siguientes:

- **Frecuencia de muestreo 10 kHz:**
  - Latencia media: 8.47582772 e3 ns
  - Desviación típica de la latencia: 2.68563047 e5 ns
- **Frecuencia de muestreo 15 kHz:**
  - Latencia media: 8.44964243 e3 ns
  - Desviación típica de la latencia: 2.67485489 e5 ns
- **Frecuencia de muestreo 20 kHz:**
  - Latencia media: 5.67483012 e3 ns
  - Desviación típica de la latencia: 1.97585501 e5 ns

Como se puede observar en los resultados obtenidos, la media de las latencias obtenida en cada una de las pruebas está muy por debajo del orden de los milisegundos por lo que se puede afirmar que es viable usar la aplicación desarrollada para RTXI en este proyecto.

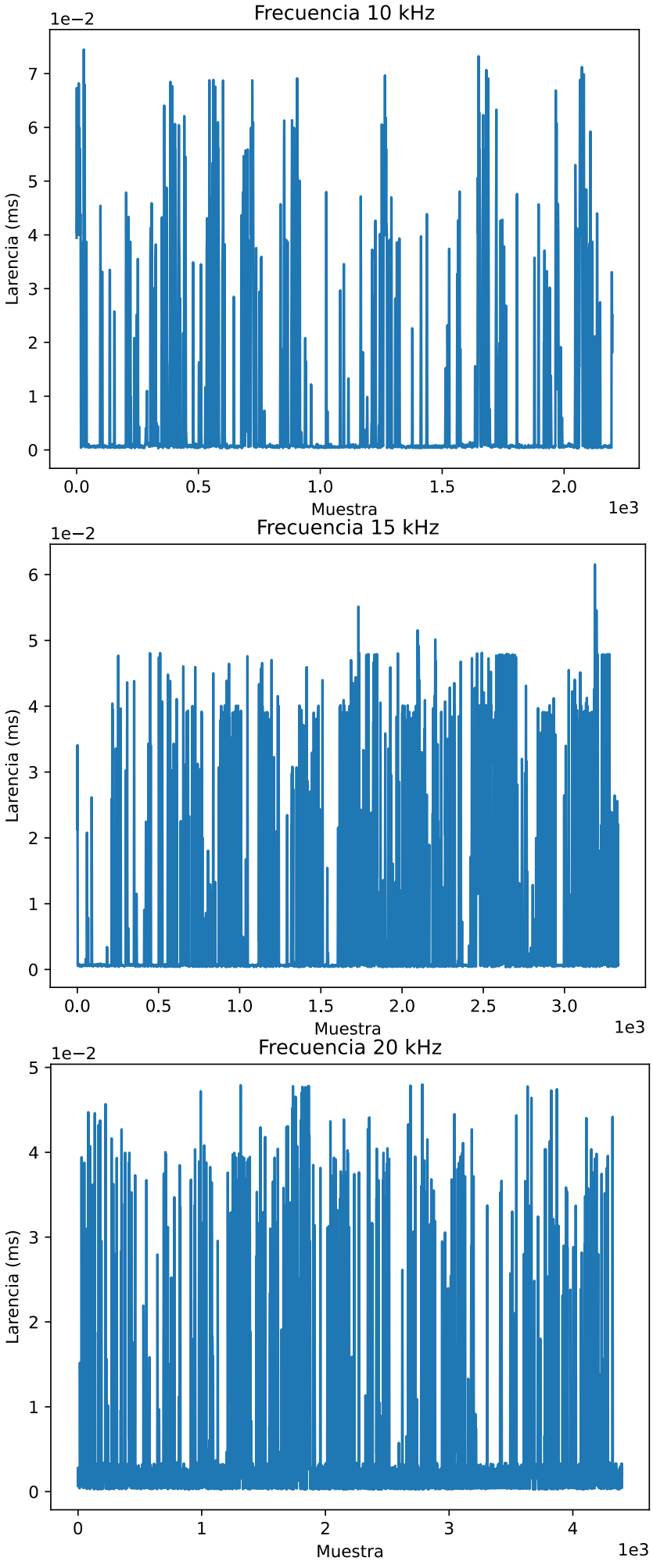

Figura 24 Gráficas de latencias (*sliding moving window*) con frecuencias de muestreo de 10 kHz, 15 kHz y 20 kHz con un tamaño de ventana de 1000 muestras y un solape de 500 muestras.

Una descripción con más detalle de la evolución temporal de las latencias durante el experimento se puede observar en la Figura 24, donde se puede ver como para cada frecuencia de muestreo los datos de las latencias obtenidas son considerablemente inferiores al orden de los milisegundos, confirmando de nuevo la viabilidad de la aplicación desarrollada.

Sin embargo, en todas las pruebas existe un porcentaje ínfimo, en torno al 0.08% que sobrepasa el milisegundo, lo que significa que se están perdiendo datos cuando se ejecutan tareas con esta latencia. Este hecho se debe a que en el generador de señales pregrabadas se carga el estímulo entero en un buffer al cual se accede en tiempo real, por lo tanto, esta información se guarda en la memoria caché del ordenador junto con otros datos. Como el estímulo cargado en estas pruebas tiene un número de datos considerable durante la ejecución de estas hay ocasiones en las que el tiempo de acceso a estos datos se dispara provocando estas latencias.

Al haber atravesado un periodo de confinamiento, este proyecto se desarrolló por completo en un sistema no preparado para el tiempo real, sin un pez eléctrico y sin el hardware necesario, por lo tanto, la realización de estas pruebas era imposible. Como ya se ha comentado, de este hecho nace el módulo generador de señales pregrabadas que pretende ser un apoyo a la hora de simular el voltaje emitido por el pez y poder comprobar el correcto funcionamiento del protocolo. Sin embargo, los datos precargados no reservan memoria dinámicamente en tiempo real a diferencia de una tarjeta de adquisición de datos, que posee librerías que se encargan de gestionarlo, por lo que el acceso a memoria puede ser más lento en algunas ocasiones, es decir, las pruebas pensadas para este proyecto estaban enfocadas a usar un pez eléctrico.

8 Conclusiones y trabajo futuro

8.1 Conclusiones

En este proyecto se ha utilizado RTXI como entorno de desarrollo en tiempo real para implementar una aplicación para el estudio de la codificación de información en el sistema nervioso, más específicamente se ha adaptado a esta tecnología el método TCDS desarrollado por el Grupo de Neurocomputación Biológica (GNB, http://www.ii.uam.es/~gnb) en [1, 2].

Este método se desarrolló para una herramienta que también se utiliza para la implementación de funcionalidad en tiempo real llamada ADClamp y que al igual que RTXI está enfocada en la adquisición y control de datos para estudios e investigaciones biológicas.

Sin embargo, ADClamp está actualmente en desuso, no tiene mantenimiento de ningún tipo, sus paquetes software están desactualizados y es difícil de instalar. Por otro lado, RTXI sigue recibiendo soporte software, hay numerosos grupos de investigación que hacen uso de esta herramienta por lo que existe una potencial expansión del software desarrollado en este proyecto ya que cualquier grupo que use RTXI puede instalar los módulos que se han implementado con mucha facilidad.

Además, RTXI aporta un gran número de ventajas a la hora de realizar proyectos con esta herramienta, como su arquitectura modular, todos los módulos del sistema que tiene están enfocados a la neurociencia y la biología, su fácil uso e instalación, la facilidad con la que puedes construir tus propios módulos, etc.

Es por estas diferencias que hacen de RTXI un entorno útil e interesante al que portar este método para poder realizar este tipo de estudios.

En el desarrollo de este proyecto se han implementado tres módulos, un generador de señales pregrabadas, el módulo *WORDS* y un generador de estímulos. Cabe destacar que este software es libre (https://github.com/GNB-UAM/tcds-rtxi).

Uno de los módulos desarrollados, el generador de señales, es consecuencia directa del reciente confinamiento por el que hemos pasado al no poder utilizar un pez eléctrico como fuente de generación de las señales, debido a esto se implementó este módulo, el cual puede generar cualquier tipo de señal leída por fichero con el objetivo de poder simular el voltaje emitido por el campo eléctrico de un pez para poder comprobar el correcto funcionamiento del protocolo desarrollado.

Para analizar el voltaje adquirido se implementó el módulo *WORDS*, cuya principal función es adaptar el método TCDS [1, 2] para RTXI, de tal forma que analiza la señal y le comunica al generador de estímulos cuando tiene que emitirlos.

Por último, el generador de estímulos es el encargado de emitir una señal precargada mediante la lectura de un fichero para enviar el estímulo en cuanto el módulo *WORDS* se lo comunique. De esta forma, haciendo uso de los tres módulos se ha diseñado una herramienta en tiempo real para el análisis del procesamiento de la información en el sistema nervioso en ciclo cerrado y en el contexto de los peces eléctricos.

Todas las ventajas que aporta RTXI no son útiles si no se pueden realizar experimentos haciendo uso de esta aplicación, es decir, es necesario saber que las tareas de los módulos desarrollados en este proyecto que se ejecutan en tiempo real deben tener unas latencias mínimas para poder procesar la información con exactitud. Es por esto por lo que se han realizado distintas tomas de latencias durante la ejecución de los experimentos en tiempo real. Para esta toma de latencias se utilizaron todos los módulos que se precisan para un estudio completo del procesamiento de información del sistema nervioso en los peces eléctricos en el contexto de la electrorrecepción, así como el hardware que también es necesario para ello. De esta manera se realizaron pruebas de latencias para frecuencias de 10 kHz, 15 kHz y 20 kHz, un rango más que suficiente para cubrir este tipo de experimentos. Como resultado de estas pruebas se obtuvo una media de latencias bastante inferior al orden de los milisegundos por lo que se puede afirmar que es viable el uso RTXI para el desarrollo de este tipo de experimentos con los medios disponibles en el laboratorio (GNB, Escuela Politécnica Superior, Universidad Autónoma de Madrid).

Llegados a este punto cabe destacar que un porcentaje ínfimo de las latencias obtenidas en estas pruebas, el 0,08 no cumple las restricciones requeridas. La principal causa de este hecho es que al precargar como estímulo un gran número de datos, en ocasiones el tiempo de acceso a la memoria es tan grande que provoca estas latencias, sin embargo, como trabajo de futuro, se investigará con más detalle el origen de estas latencias. Debido al confinamiento que hemos vivido ha sido imposible hacer uso de un pez eléctrico real para

realizar estas pruebas dando lugar a estas latencias que no comprometen la viabilidad del proyecto.

Para poder concluir con este proyecto, era necesario comprobar que todos los módulos externos al sistema de RTXI funcionan correctamente, para ello se realizaron algunos experimentos. Para empezar, se probó que los módulos propios del sistema podían ser utilizados con el hardware disponible en el laboratorio, es decir, que se podía hacer un pequeño experimento en el que RTXI utilizará la tarjeta de adquisición de datos del laboratorio, de esta forma se comprobó el correcto funcionamiento tanto del hardware como del software más básico del proyecto. A continuación, se realizaron distintos experimentos que confirmaron que cada uno de los módulos desarrollados en este proyecto realizaban la funcionalidad para la cual se les había diseñado.

8.2 Trabajo futuro

Tras el desarrollo de este trabajo se han encontrado diversos aspectos a desarrollar como trabajo futuro. Los más destacables son:

- En la funcionalidad relacionada con la estimulación, existen una gran cantidad de posibilidades que pueden desarrollarse como trabajo futuro para este proyecto. El módulo generador de estímulos puede ampliar notoriamente sus funciones, aumentando el número de estímulos que precarga y emitiéndolos en función de la señal que recibe. También podría encargarse de configurar los módulos generador de ruido y generador de señales con el objetivo de usarlos como salida del experimento y no como entrada existiendo así la posibilidad de producir una gran cantidad de estímulos distintos.
- Ampliación del protocolo: Como se ha mencionado anteriormente el protocolo TCDS tiene muchas formas de ampliarse y muchas decisiones que se pueden tomar a la hora de analizar el comportamiento del sistema. Por ejemplo, se puede retrasar la estimulación un tiempo fijo o variable una vez se ha detectado la palabra. Este punto es interesante ya que deja abierta diversas vías de estudio y puede ser muy útil para conocer más sobre el sistema nervioso del organismo.
- Diseñar e implementar una batería de *tests* automatizados para comprobar el funcionamiento del protocolo implementado.
- Uno de los principales problemas que ha estado presente en la toma de medidas de latencias de la herramienta RTXI es la falta de una referencia temporal para hacer un análisis exacto de la ejecución de tareas en tiempo real. En el futuro se puede realizar este análisis mediante el uso de un osciloscopio como referencia temporal [45].
- Como se ha analizado en las pruebas de latencias, precargar un estímulo con una cantidad de datos considerable puede hacer que el tiempo de acceso a la caché provoque latencias no deseables para el tiempo real, por ello, como trabajo a futuro existe la posibilidad de poder mejorar la forma en la que se carga el estímulo, es decir, reservar memoria de forma dinámica y en tiempo real para estos datos evitando que se den lugar estas latencias.
- Otro trabajo de futuro de gran utilidad consiste en realizar pruebas de latencia para distintas cargas del sistema, de esta forma se puede comprobar cuanta carga aguanta el sistema manteniendo la precisión requerida.

Referencias

[1] Lareo, Á. (2014). Estudio del procesamiento de información en peces eléctricos mediante técnicas lazo cerrado de estimulación en tiempo real. Trabajo Fin de Máster. Departamento de Ingeniería Informática (UAM).

[2] Lareo, Á., Forlim, C., Pinto, R., Varona, P., & Rodríguez, F. (2016). Temporal Code-Driven Stimulation: Definition and Application to Electric Fish Signaling. *Frontiers in Neuroinformatics*. https://doi.org/10.3389/fninf.2016.00041

[3] Chamorro, P., Muñiz, C., Levi, R., Arroyo, D., Rodríguez, F., & Varona, P. (2012). Generalization of the Dynamic Clamp concept in neurophysiology and behavior. *PLoS ONE, 7*(7), e40887. https://doi.org/10.1371/journal.pone.0040887.

[4] Muñiz, C., Levi, R., Benkrid, M., Rodríguez, F., & Varona, P. (2008). Real-time control of stepper motors for mechano-sensory stimulation. *Journal of Neuroscience Methods, 172*(1), 105-111. https://doi.org/10.1016/j.jneumeth.2008.04.017

[5] Muñiz, C., Arganda, S., Rodríguez, F., & Polavieja, G. (2005). Realistic Stimulation Through Advanced Dynamic-Clamp Protocols. *Lect Notes Comput Sc*, 95-105. https://doi.org/10.1007/11499220_10

[6] Folim, C., Pinto, R., Varona, P., & Rodríguez, F. (2015). Delay-Dependent Response in Weakly Electric Fish under Closed-Loop Pulse Stimulation. *PLOS ONE, 10*(10), e0141007. https://doi.org/10.1371/journal.pone.0141007

[7] Varona, P., Guardeño, D. A., Nowotny, T., & Rodríguez, F. (2016). Online event detection requirements in closed-loop neuroscience. *Closed Loop Neuroscience*, 81-91. https://doi.org/10.1016/B978-0-12-802452-2.00006-8

[8] Fernández-Varga, J., Pfaff, H. U., Rodríguez, F., & Varona, P. (2013). Assisted closed-loop optimization of SSVEP-BCI efficiency. *Frontiers in Neural Circuits, 7*, 27. https://doi.org/10.3389/fncir.2013.00027

[9] Robinson, H. P. (1991). Kinetics of synaptic conductances in mammalian central neuron. *Neuroscience Research Supplements, 16*, VI. https://doi.org/10.1016/0921-8696(91)90611-p

[10] Robinson, H. P. (1993). Injection of digitally synthesized synaptic conductance transients to measure the integrative properties of neurons. *Journal of neuroscience methods, 49*(3), 157-165. https://doi.org/10.1016/0165-0270(93)90119-c

[11] *Papers*. (s.f.). Recuperado el 22 de Julio de 2020, de http://rtxi.org/papers/

[12] Weiskopf, N., Scharnowski, F., Veit, R., Goebel, R., Birbaumer, N., & Mathiak, K. (2004). Self-regulation of local brain activity using real-time functional magnetic resonance imaging (fMRI). *Journal of Physiology-Paris, 98*(4-6), 357-373. https://doi.org/10.1016/j.jphysparis.2005.09.019

[13] Huang, H., Wolf, S. L., & He, J. (2006). Recent developments in biofeedback for neuromotor rehabilitation. *Journal of neuroengineering and rehabilitation, 3*(1), 11. https://doi.org/10.1186/1743-0003-3-11

[14] Lünenburger, L., Colombo, G., & Riener , R. (2007). Biofeedback for robotic gait rehabilitation. *Journal of neuroengineering and rehabilitation, 4*(1), 1-11. https://doi.org/10.1186/1743-0003-4-1

[15] Nestoriuc, Y., Martin, A., Rief, W., & Andrasik , F. (2008). Biofeedback treatment for headache disorders: a comprehensive efficacy review. *Applied psychophysiology and biofeedback, 33*(3), 125-140. https://doi.org/10.1007/s10484-008-9060-3

[16] DeCharms , R. C. (2008). Applications of real-time fMRI. *Nature Reviews Neuroscience, 9*(9), 720-729. https://doi.org/10.1038/nrn2414

[17] *What Is Real-time Computing?* (06 de 12 de 2014). Recuperado el 18 de Julio de 2020. http://rtxi.org/docs/tutorials/2014/12/06/what-is-real-time-computing/

[18] *User Manual*. (s.f.). Recuperado el 20 de Julio de 2020, de http://rtxi.org/docs/manual/

[19] Sharp, A. A., O'Neil, M. B., Abbott, L. F., & Marder, E. (1993). Dynamic clamp: computer-generated conductances in real neurons. *Journal of neurophysiology, 69*(3), 992-995. https://doi.org/10.1152/jn.1993.69.3.992

[20] Batera, R. J., Wilson, C. G., Rinzel, J., & Smith, J. C. (1999). Fast model-reference current injection using RT-Linux. *Proceedings of the First Joint BMES/EMBS Conference. 1999. 2*, págs. 876--vol. Atlanta, USA: IEEE Engineering in Medicine and Biology. https://doi.org/10.1109/IEMBS.1999.804031

[21] Milescu, L. S., Yamanishi, T., Ptak, K., Mogri, M. Z., & Smith, J. C. (2008). Real-Time Kinetic Modeling of Voltage-Gated Ion Channels Using Dynamic Clamp. *Biophysical Journal, 95*(1), 66-87. https://doi.org/10.1529/biophysj.107.118190

[22] Barón-Esquivias, G., Moya-Mitjans, A., Martinez-Alday, J., Ruiz-Granell, R., Lacunza-Ruiz, J., Garcia-Civera, R., . . . Morillo, C. A. (2020). Impact of dual-chamber pacing with closed loop stimulation on quality of life in patients with recurrent reflex vasovagal syncope: results of the SPAIN study. *EP Europace, 22*(2), 314-319. https://doi.org/10.1093/europace/euz294

[23] Buccino, A. P., Lepperød, M. E., Dragly, S.-A., Häfliger, P., Fyhn, M., & Hafting, T. (2018). Open source modules for tracking animal behavior and closed-loop stimulation based on Open Ephys and Bonsai. *Journal of Neural Engineering, 15*(5). https://doi.org/10.1088/1741-2552/aacf45

[24] Angotzi, G. N., Boi, F., Zordan, S., Bofanti, A., & Vato, A. (2014). A programmable closed-loop recording and stimulating wireless system for behaving small laboratory. *Scientific Reports, 4*(1). https://doi.org/10.1038/srep05963

[25] Pineda, R., Beattie, C. E., & Hall, C. W. (2012). Closed-loop neural stimulation for pentylenetetrazole-induced seizures in zebrafish. *Disease Models & Mechanisms, 6*(1), 64-71. https://doi.org/10.1242/dmm.009423

[26] MacIver MA (2001) The computational neuroethology of weakly electric fish: Body modeling, motion analysis, and sensory signal estimation. Ph.D. Dissertation, University of Illinois at Urbana Champaign. UMI, Ann Arbor, MI, USA.

[27] Mileva, G., Zysman, D., Groothuis, S., & Lewis, J. E. (2008). In vitro studies of closed-loop feedback and electrosensory processing in Apteronotus leptorhynchus. *Journal of Physiology-Paris, 102*(4-6), 173-180. https://doi.org/10.1016/j.jphysparis.2008.10.012

[28] Forlim, C., & Pinto, R. (2014). Automatic Realistic Real Time Stimulation/Recording in Weakly Electric Fish: Long Time Behavior Characterization in Freely Swimming Fish and Stimuli Discrimination. *PLoS ONE, 9*(1), e84885. https://doi.org/10.1371/jour

[29] García-García, V. H. (s.f.). Desarrollo de una “toolbox” para condicionar el comportamiento de peces eléctricos, basado en la codificación de las señales eléctricas.

[30] *Xenomai Home*. (s.f.). Recuperado el 10 de Julio de 2020, de https://gitlab.denx.de/Xenomai/xenomai/-/wikis/home

[31] *RTAI Home*. (s.f.). Recuperado el 12 de Julio de 2020, de https://www.rtai.org/
[32] Von der Emde , G. (2006). Non-visual environmental imaging and object detection through active electrolocation in weakly electric fish. *Journal of Comparative Physiology*, 601-612. https://doi.org/10.1007/s00359-006-0096-7
[33] Bullock, T. H., Hopkins, C. D., Popper, A. N., & Fay, R. R. (2006). *Electroreception.* New York: Springer Publishing. https://doi.org/10.1007/0-387-28275-0
[34] Carlson, B. A., Sisneros, J. A., Poppe, A. N., & Fay, R. R. (2019). *Electroreception: Fundamental Insights from Comparative Approaches.* New York: Springer. https://doi.org/10.1007/978-3-030-29105-1
[35] Kramer, B. (1996). *Electroreception and communication in fishes. Progress in Zoology.* Stuttgart: Gustav Fisher Verlag.
[36] Baker, C. V., Modrell, M. S., & Gillis, J. A. (2013). The evolution and development of vertebrate lateral line electroreceptors. *Journal of Experimental Biology*, 2515-2522. https://doi.org/10.1242/jeb.082362
[37] Dunlap, K. D. (2012). Hormonal and Body Size Correlates of Electrocommunication. *Journal of Fish Biology, 81*(7), 2235-2254. https://doi.org/10.1111/j.1095-8649.2012.03448.x
[38] Gebhardt, K., Böhme, M., & Von der Emde, G. (2012). Electrocommunication behaviour during social interactions in two species of pulse-type weakly electric fishes (Mormyridae). *Journal of Fish Biology, 81*(7), 2235-2254. https://doi.org/10.1111/j.1095-8649.2012.03448.x
[39] Dunlap, K. D., Thomas, P., & Zakon, H. H. (1998). Diversity of sexual dimorphism in electrocommunication signals and its androgen regulation in a genus of electric fish, Apteronotus. *Journal of Comparative Physiology, 183*(1), 77-86. https://doi.org/10.1007/s003590050236
[40] Macadar, O., & Silva, A. (2007). Electric communication in South American gymnotiform fishes. *Revista Latinoamericana de Psicología, 39*(1), 31-45.
[41] Kramer, B. (1990). *Electrocommunication in teleost fishes: behavior and experiments.* Berlín: Springer-Verlag. https://doi.org/10.1007/978-3-642-84026-5
[42] Bennett, M. V. (1971). Electroreception. *Fish Physiology, 5*, 493-574. https://doi.org/10.1016/S1546-5098(08)60052-7
[43] Von der Emde, G. (1999). Active electrolocation of objects in weakly electric fish. *Journal of Experimental Biology, 202*, 1205-1215.
[44] *RTXI Architecture*. (s.f.). Recuperado el 15 de Julio de 2020, de http://rtxi.org/docs/tutorials/2014/12/06/rtxi-architecture/
[45] Amaducci, R., Reyes-Sánchez, M., Elices, I., Rodríguez, F., & Varona, P. (2019). RTHybrid: A Standardized and Open-Source Real-Time Software Model Library for Experimental Neuroscience. *Frontiers in Neuroinformatics, 13*. https://doi.org/10.3389/fninf.2019.00011
[46] *¿Qué es Adquisición de Datos?- Productos y Servicios - National Instruments*. (s.f.). Recuperado el 15 de Julio de 2020, de https://web.archive.org/web/20071023023101/http://www.ni.com/dataacquisition/esa/whatis.htm
[47] *RTXI/tutorials*. (s.f.). Recuperado el 20 de Julio de 2020, de https://github.com/RTXI/tutorials/wiki/MyPluginGUI-RTXI-Functions

Glosario

ADCLAMP	Advanced Dynamic Clamp
API	Application Programming Interface
DAQ	Data Acquisition Card
FIFO	First In First Out
GNB	Grupo de Neurocomputación Biológica
GUI	Graphical User Interface
ID	Identification
IPI	Inter Pulse Interval
MRCI	Model Reference Current Injection
OE	Órgano Eléctrico
PCI	Peripheral Component Interconnec
POSIX	Portable Operating System Interface for X
PXI	PCI Extensions for Instrumentation
RTAI	Real-Time Application Interface
RTDM	Real Time Driver Model
RTDOC	Real Time Dynamical Observer and Controller
RTLDC	Real-Time Linux Dynamic Clamp
RTOS	Real Time Operating System
RTXI	Real-Time eXperiment Interface
SO	Sistema Operativo
STR	Sistema en Tiempo Real
TCDS	Temporal Code-Driven Stimulation

Anexos

A. Adquisición de datos

La adquisición de datos es el proceso de tomar muestras del mundo real, de manera automatizada y desde recursos de medidas analógicas, para generar datos que puedan ser utilizados por un ordenador, de esta manera se pueden detectar señales físicas, convertirlas en potencial eléctrico y digitalizarlas para que un ordenador pueda procesarlas como un conjunto de datos. Para llevar a cabo este proceso es indispensable el uso de las tarjetas de adquisición de datos (DAQ) [46].

B. Archivos HDF5

RTXI dispone de un módulo del sistema denominado *Data Recorder*, cuyo principal objetivo es guardar las salidas obtenidas en tiempo real de los distintos módulos cargados.

Los archivos generados por este módulo están en formato HDF5, un formato de datos binario y extensible diseñado para la captura de datos complejos, además, presenta una estructura jerárquica (Figura 25) que le permite acceder a fragmentos de archivos sin cargar todo en memoria [18].

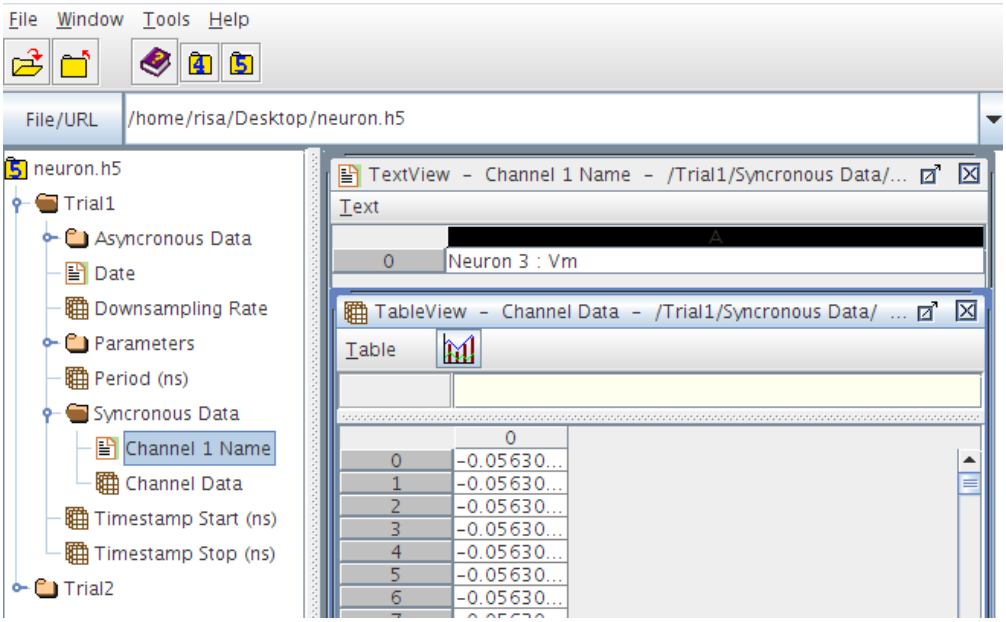

Figura 25 Ejemplo formato HDF5.

C. Analogy

Analogy es un proyecto software gratuito incluido en Xenomai que proporciona servicios de gestión de dispositivos de adquisición de datos en tiempo real. Además, Analogy es un proyecto basado en otro, Comedi, que es un marco para las tarjetas de adquisición de datos desarrollado en su origen por Dacid Schleef con el objetivo de poder usar una

interfaz gráfica sobre muchas y diferentes tarjetas, pudiendo así tomar mediciones y llevar un control de los datos adquiridos. Estas librerías se usan principalmente en Linux con RTAI [30].

El proyecto Analogy está basado en el *"Real Time Driver Model"* cuyo objetivo es unificar las interfaces para desarrollar controladores de dispositivos y aplicaciones asociadas a Linux. Por lo tanto, gracias al RTDM implementado en Xenomai y al determinismo en tiempo real ya sea en el usuario como en el núcleo, es posible que una tarea que se esté ejecutando en tiempo real de Xenomai basada en cualquier máscara (como puede ser POSIX) puede usar Analogy sin restricciones.

Al estar desarrollado sobre Xenomai, RTXI usa la interfaz de Analogy a la hora de adquirir datos en tiempo real de las tarjetas de adquisición de datos, admitiendo así una larga lista de tarjetas PCI / PXI de *National Instrument* además de ser compatible con la placa Sensory s526.

D. RTAI

RTAI (*Real Time Application Interface*) consiste en una implementación de Linux basada en RTLinux, es decir, esta herramienta se basa en el núcleo de Linux con la cualidad de poder hacerlo completamente requisable (*preemtable*), tratándolo como si fuera una tarea de menor prioridad. La estructura de esta herramienta se puede ver en la Figura 26.

La comunicación entre procesos está gestionada en RTAI por los siguientes métodos [31].

- RTAI consta de una API de semáforos para la sincronización entre tareas.
- Existe una memoria compartida como bloque común que puede ser leída o escrita por cualquier proceso facilitando la comunicación entre la parte *real-time* con la interfaz de usuario.
- El método más flexible es el uso de las *mailboxes*. Cualquier proceso puede enviar y recibir desde una *mailbox*, en donde se almacenan los mensajes hasta un límite predefinido. Se pueden crear tantas como sean necesarias.
- Por último, existen FIFOs propias de RTXI. A diferencia de las FIFOs convencionales y propias de Linux, RTAI puede lanzar señales cuando hay eventos en ellas, por ejemplo, una escritura, de tal forma que se pueden manejar estas señales a conveniencia. Además, pueden coexistir múltiples lectores y escritores en una FIFO y pueden ser localizadas mediante identificadores simbólicos.

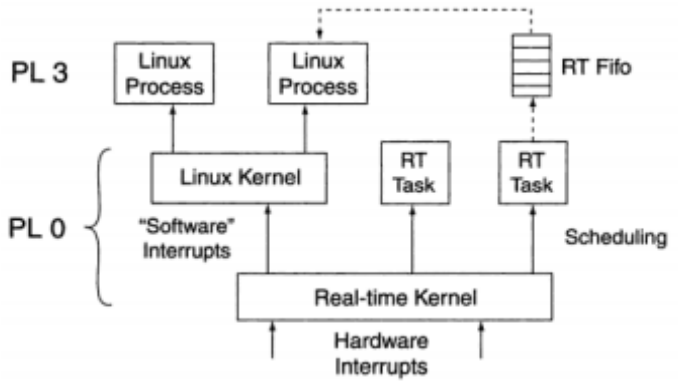

Figura 26 Arquitectura de RTAI. Figura extraída de [1].

E. Xenomai

Xenomai es un Proyecto de software libre desarrollado por ingenieros con una gran cantidad de experiencia en el marco de los procesos en tiempo real para el sistema operativo Linux.

El objetivo principal de Xenomai consiste en ayudar a migrar aplicaciones de ámbito industrial de sistemas en tiempo real a Linux.

Xenomai se lanzó en agosto del 2001, en el 2003 se fusiona completamente con el proyecto RTAI para producir una plataforma de software libre en tiempo real. Aunque terminó separándose en el 2005 siguiendo su camino de forma independiente.

Xenomai tiene una estructura de núcleo doble, es decir, implementa un *microkernel* que se ejecuta en paralelo con un *kernel* de Linux separado, proporcionando así un soporte generalizado para los desarrolladores.

Xenomai tiene dos arquitecturas distintas con las que se puede trabajar. [30].

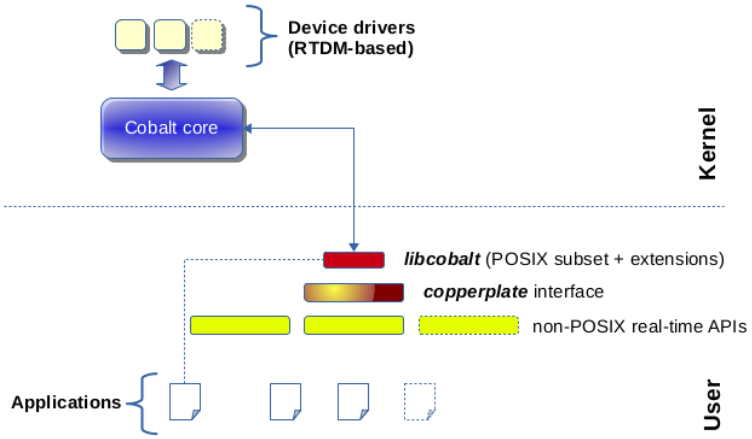

Figura 27 Doble núcleo Xenomai. Figura extraída de [30].

Al combinar el núcleo de Linux con uno en tiempo real ambos se ejecutan al mismo tiempo. En su arquitectura (Figura 27) existe pequeña extensión denominada ***Cobalt*** que está integrada en el núcleo de Linux y se encarga de gestionar todas las tareas críticas de tiempo, como por ejemplo el manejo de las interrupciones o la programación de subprocesos en tiempo real, teniendo así más prioridad sobre las actividades del núcleo nativo.

En esta configuración de doble núcleo todas las API de RTOS Xenomai aportan una interfaz con el núcleo Cobalt siendo así las únicas capaces de utilizar el tiempo real.

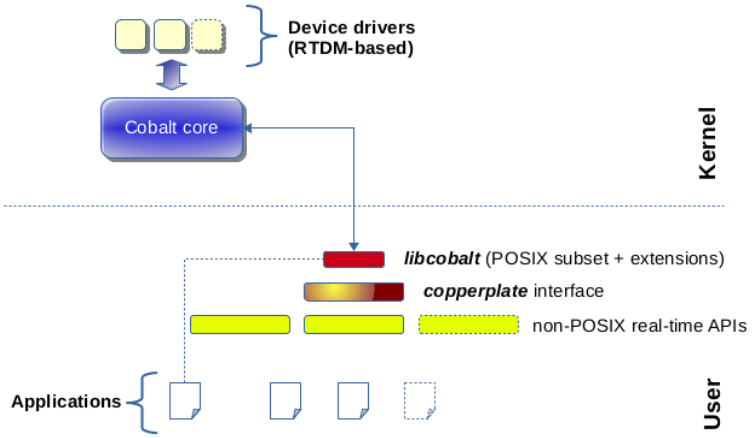

Figura 28 Núcleo único Xenomai. Figura extraída de [30].

Confiando las capacidades de tiempo real al núcleo nativo de Linux, dando lugar al núcleo ***Mercury*** (Figura 28).

En esta configuración de núcleo único todas las API RTOS que no son de POSIX que Xenomai proporciona se emulan sobre la biblioteca de subprocesos nativa.

Las principales diferencias entre Xenomai y RTAI derivan de los objetivos a los que van dirigidos ambos proyectos. Si bien RTAI busca técnicamente que los procesos que se ejecutan bajo su arquitectura tengan las latencias más bajas, Xenomai, aunque considera también este hecho, se centra mucho en la extensibilidad, la portabilidad y la mantenibilidad como objetivos prioritarios.

F. ADClamp

ADClamp (*Advanced Dynamic Clamp*)[1] es una herramienta desarrollada por el GNB (Grupo de Neurocomputación Biológica) de la universidad autónoma de Madrid. Esta herramienta permite la interacción bidireccional entre sistemas bilógicos y no biológicos y ha sido utilizada para diversos experimentos de interacciones de circuito cerrado en tiempo real por objetivos con microinyecciones de drogas, dispositivos mecánicos y estimulación impulsada por eventos de vídeo [3].

Esta herramienta ha sido desarrollada con el objetivo de implementar y diseñar métodos de estimulación en ciclo cerrado en el concepto del observador/controlador dinámico en tiempo real (RTDOC), de tal forma que se establece una comunicación bidireccional retroalimentada entre el ente artificial y el sistema biológico.

Los protocolos desarrollados en esta herramienta se ejecutan en una extensión en tiempo real del *kernel* de Linux conocida como RTAI que garantiza que las operaciones especificadas por el software se ejecuten en tiempo real, mientras se mantiene el rendimiento general del sistema operativo.

Además, esta herramienta utiliza un proyecto de código abierto denominado *COontrol and MEasurement Device Driver* que proporciona herramientas, controladores y librerías para gestionar una gran variedad de componentes relacionados con la adquisición de datos. La mayoría de las DAQ utilizadas en experimentos electrofisiológicos son compatibles con estas bibliotecas. Además, es un software fácil de usar al incluir un GUI personalizable programada con la biblioteca Qt de C ++.

Como ya se ha mencionado RTAI puede gestionar procesos tanto en tiempo real como en espacio de usuario donde se puede trabajar con aplicaciones ordinarias de Linux de uso general. En ADClamp existen dos procesos que se ejecutan simultáneamente, el proceso en tiempo real escrito en C y la GUI que se ejecuta en el espacio de usuario, por lo tanto, son dos procesos independientes que trabajan en paralelo y que necesitan que exista un flujo de datos entre ambos para que se produzca el intercambio de información. Por ello ADClamp cuenta con dos FIFOs desarrolladas específicamente para que sea posible esta comunicación. En la primera de ellas el proceso en tiempo real escribe el voltaje obtenido por la DAQ y la GUI necesita un temporizador para poder acceder a esta información que puede ser configurado por el usuario para adaptar el rendimiento de la aplicación. La otra FIFO se utiliza para modificar el comportamiento del proceso ejecutado en tiempo real mediante la modificación de los parámetros de la GUI. Por último, la topología de la red se almacena en una memoria compartida accesible por ambos procesos [1].

[1] http://www.ii.uam.es/~gnb/adclamp/

Esta herramienta trabaja a una frecuencia de 15 KHz, por lo que el periodo de muestreo es de 66.6 ns, más que suficiente para el estudio del comportamiento de los peces eléctricos.

G. PluguinGui

Los módulos construidos en RTXI por los desarrolladores siguen un patrón prediseñado que la propia herramienta provee. Este patrón define la estructura del módulo y sirve como guía para el desarrollo de la funcionalidad de los proyectos en tiempo real.

La clase base que se implementa en RTXI se denomina *MyPluginGUI* y consta de las siguientes partes [47].

Definición de variables

```
static DefaultGUIModel::variable_t vars[] =
  {
    { "GUI label", "Tooltip description", DefaultGUIModel::PARAMETER
        | DefaultGUIModel::DOUBLE, },
    { "A State", "Tooltip description", DefaultGUIModel::STATE, }, };

static size_t num_vars = sizeof(vars) / sizeof(DefaultGUIModel::variable_t);
```

Figura 29 Definición de variables *PluguinGui.* Figura extraída de [47].

En esta parte del código es donde se pueden definir las entradas y salidas que formarán las conexiones del módulo, así como los parámetros, eventos y estados a los que se podrá acceder y modificar a través de la interfaz del código y que se generarán cuando se construya la misma (Figura 29).

Constructor de la clase

```
MyPluginGUI::MyPluginGUI(void) :
  DefaultGUIModel("MyPluginGUI with Custom GUI", ::vars, ::num_vars)
{

  QWhatsThis::add(this, "<p><b>MyPluginGUI:</b><br>QWhatsThis description.</p>");
  createGUI(vars, num_vars); // this is required to create the GUI
  update( INIT); // this is optional, you may place initialization code directly into the constructor
  refresh(); // this is required to update the GUI with parameter and state values
}
```

Figura 30 Constructor de la clase *PluguinGui.* Figura extraída de [47].

Cada vez que se realiza una instancia del módulo se llama al constructor, aquí es donde se puede añadir una descripción general de la funcionalidad a desarrollar, se inicializan todos los parámetros antes de que tenga lugar la ejecución en tiempo real y se construye la interfaz gráfica del módulo desarrollada en Qt (Figura 30).

Métodos

Realmente es en los métodos donde se específica cómo debe ser el funcionamiento de la tarea cuando se ejecuta en tiempo real. Esta es la parte más crucial del módulo ya que de ella depende que la tarea lleve a cabo su cometido correctamente. Los dos métodos más importantes para desarrollar un módulo en RTXI son ***execute*** y ***update***.

En el método *execute* (Figura 31), cuando el sistema en tiempo real está ejecutándose, toma muestras con una periodicidad determinada por los ajustes de éste, en cada uno de estos periodos se llama a esta función en cada uno de los módulos que se estén ejecutando en tiempo real. Esto permite a la tarea modificar y tratar datos en tiempo real mediante el uso de inputs y outputs declarados en la matriz de variables que procesan y producen las señales en tiempo real.

A su vez, se pueden definir funciones con distintos objetivos que se vayan a ejecutar en tiempo real siempre y cuando sean llamadas dentro de este método.

```
void
MyPluginGUI::execute(void)
{
  return;
}
```

Figura 31 Método *execute PluguinGui.* Figura extraída de [47].

El método *update* (Figura 32) es el mecanismo que se utiliza para captar los datos modificados en la parte de usuario, a través de los valores mostrados en la interfaz gráfica mientras se está ejecutando el módulo en tiempo real.

Cada vez que el sistema detecta un evento en la interfaz gráfica como un cambio del valor de un parámetro o el presionar un botón, se coloca una bandera a la que se asocia el tipo de cambio que ha tenido lugar, de esta forma el desarrollador puede especificar como actuará el módulo que está desarrollando en función del tipo de evento que haya tenido lugar durante la ejecución de la tarea en tiempo real.

Esta función contiene una estructura lógica denominada como *switch* implementada en C y en otros muchos lenguajes. De esta forma para cada valor de la bandera el usuario puede especificar el tipo de respuesta que tiene lugar.

```
void
MyPluginGUI::update(DefaultGUIModel::update_flags_t flag)
{
  switch (flag)
    {
  case INIT:
    period = RT::System::getInstance()->getPeriod() * 1e-6; // ms
    setParameter("GUI label", some_parameter);
    setState("A State", some_state);
    break;
  case MODIFY:
    some_parameter = getParameter("GUI label").toDouble();
    break;
  case UNPAUSE:
    break;
  case PAUSE:
    break;
  case PERIOD:
    period = RT::System::getInstance()->getPeriod() * 1e-6; // ms
    break;
  default:
    break;

    }
```

Figura 32 Método *update PluguinGui.* Figura extraída de [47].

El código base de la herramienta tiene cinco claros casos diferenciados:

- *INIT*: la bandera se activa cuando se inicializa el módulo, es un caso opcional y que solo se llama cuando se inicializa el constructor de la clase.
- *MODIFY*: Se activa cuando el usuario ha realizado un cambio de los parámetros en la interfaz gráfica. Para que la *flag* se active el usuario debe modificar al menos un parámetro y seleccionar el botón *"Modify"*.
- *UNPAUSE*: Indica que el módulo puede continuar con su ejecución después de haber sido pausado.
- *PAUSE*: Esta bandera se activa cuando el usuario pausa el módulo a través de la interfaz gráfica.

H. Gnathonemus Petersii y la electrorrecepción

El sistema biológico en el que se basa este proyecto es el *Gnathonemus Petersii* (Figura 33) también conocido como el pez elefante. La distribución geográfica de esta especie está diversificada en torno a la parte central y occidental de África. Habita en las partes más profundas de los ríos como el Congo, Níger, Ogun, Wouri, etc. La principal característica de estos hábitats es la reducida visibilidad que hay en ellas debido a que son aguas lodosas y lentas donde abundan una gran cantidad de especies vegetales acuáticas.

Debido a este motivo, este sistema biológico utiliza un mecanismo sensorial mediante el cual es capaz de obtener la información que necesita del medio a través del uso de campos eléctricos, también conocido como electrorrecepción.

Figura 33 *Gnathonemus Petersii.*

I. Electrorrecepción

Cuando pensamos en cómo perciben estímulos en su ecosistema las especies que pertenecen al reino animal tendemos con frecuencia a pensar desde un punto de vista antropocéntrico, es decir, pensamos en sentidos como el gusto, el tacto, el olfato…

Sin embargo, las formas en las que los animales son capaces de extraer información de los ecosistemas de los que forman parte es muy diversa. Existen especies que son capaces de detectar olores en un espectro muy amplio, un rango mucho mayor de frecuencias de sonidos, sensibilidad a corrientes de aire, detección del espectro infrarrojo e incluso son capaces de detectar las vibraciones del suelo. El mecanismo que se aborda en este trabajo, la electrorrecepción, es una de las formas que usan algunos animales para obtener esta información y una de las más interesantes.

Existen dos tipos de electrorrecepción, la activa y la pasiva. Un ejemplo de electrorrecepción pasiva son algunos tiburones, capaces de detectar cualquier campo eléctrico emitido por otros organismos en su entorno, es decir, solo son capaces de detectarlos, no constan de un OE (órgano eléctrico) con el que pueda emitirlos.

Por otro lado, existen especies capaces de generar señales eléctricas y que además tienen sensores que detectan campos eléctricos, es decir, tienen una electrorrecepción activa. Hay algunos sistemas biológicos que usan este órgano con el fin de inmovilizar a sus presas mediante descargas como las anguilas eléctricas, sin embargo, en este proyecto se estudia una especie que emite campos eléctricos con el fin de obtener información del medio que le rodea cuando hay poca visibilidad.

El principal descubridor y precursor de la electrorrecepción activa fue el zoólogo H. W. Lissman durante los años 50, desde este momento han sido numerosos los avances y descubrimientos que han tenido lugar en el estudio de este tipo de sistema sensorial.

Hace aproximadamente entre 200 y 300 millones de años dos especies evolutivas independientes los ***Mormyriformes*** africanos y los ***Gymnotiformes*** suramericanos llegaron por separado a una misma solución obteniendo un sistema de navegación, de detección de presas y de comunicación con conespecíficos mediante campos eléctricos autogenerados.

Existe una gran diversidad dentro del nicho de los peces eléctricos, es por eso por lo que se puede hacer una distinción en función de cómo usen las descargas eléctricas que producen. Como se ha mencionado antes las anguilas eléctricas producen descargas de alto voltaje con el fin de paralizar a sus presas, los peces que usan las descargas eléctricas para este fin se les considera fuertemente eléctricos. A su vez, los peces débilmente eléctricos como es el caso del pez elefante producen impulsos eléctricos con mucho menos voltaje que utilizan para los procesos relacionados con la electrorrecepción. Gracias a estas características, este tipo de peces eléctricos son fáciles de monitorizar y permiten la implementación de técnicas de ciclo cerrado para su estudio.

Existe una excepción en el mundo de los peces de descarga débil en el mecanismo miogénico de generación de campo eléctrico, es el caso de la familia ***Apteronotidae***. En esta subespecie durante su etapa larval, generan señales eléctricas por medio de señales miogénicas, sin embargo, al alcanzar la etapa adulta generan estos campos mediante el movimiento de iones a través de fibras espinales electromotoras, es decir, señales neurogénicas.

Dejando de lado si los peces que generan campos eléctricos de señal débil son miogénicos o neurogénicos, en función del tiempo de separación entre pulsos de sus descargas eléctricas podemos categorizarlos en dos tipos, los que generan ondas y los que generan pulsos (Figura 34).

Los ondulares generan señales de larga duración con un intervalo entre pulsos o IPI (Inter Pulse Interval) corto produciendo una señal de apariencia sinusoidal con la posibilidad de modular la frecuenta y la amplitud de esta.

Por otro lado, en los peces que generan señales eléctricas pulsares se caracterizan por ser de corta duración (100 μs a 10 ms), mientras que los intervalos entre pulso son de larga duración (5ms a 600 ms) [33, 34].

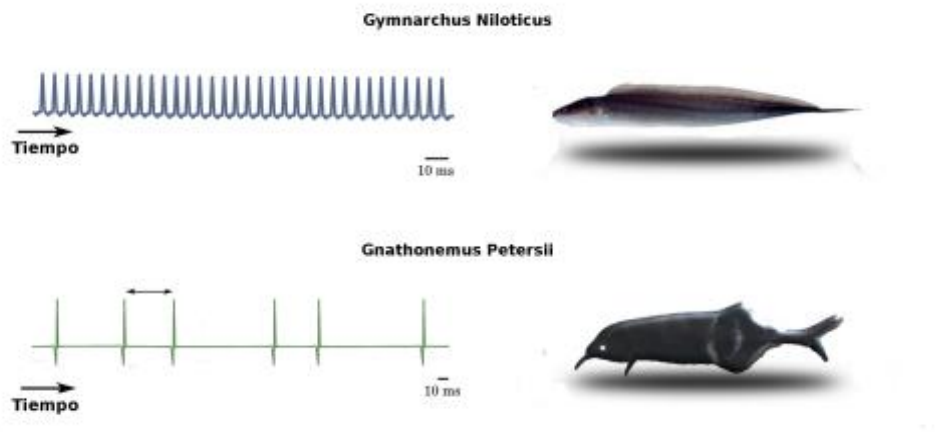

Figura 34 Tipos de señales eléctricas. Figura extraída de [1].

J. Electrolocalización y Electrocomunicación

Los encargados de detectar el campo eléctrico en los peces eléctricos de descarga débil son los receptores eléctricos localizados en la superficie del cuerpo del sistema biológico. Estos sensores son capaces de detectar diferencias de potencial entre el exterior y el interior del organismo [35].

Estos receptores son una evolución de otros mecanismos de detección que también estaban localizados en la superficie del pez, encargados de detectar otros estímulos como los movimientos que se daban en su entorno, la rotación del organismo e incluso la gravedad.

Existen dos tipos de receptores sensoriales en los peces eléctricos que emiten señales eléctricas débiles, los ampulares y los tuberosos. Estos receptores funcionan de la misma manera, son células que detectan diferencias de potencial y transforman este tipo de estímulos en mensajes químicos con neurotransmisores.

Los receptores ampulares son los más antiguos y abundan en las especies de tiburones, son extremadamente sensibles a gradientes de campos eléctricos débiles y responden a estímulos eléctricos de bajas frecuencias como los que generan sus presas [36]. Este tipo de células se encuentran bajo la superficie del cuerpo del pez y se comunican con el medio acuático.

Por otra parte, los tuberosos evolucionaron a partir de estos últimos y solo se encuentran en los peces electrogénicos como los *Mormyriformes* y los *Gymnotiformes*. Es decir, estos son los encargados de detectar el campo eléctrico emitido por el sistema biológico, de esta forma detectan las variaciones de intensidad de la señal que emiten y pueden localizar de forma exacta la posición de los distintos elementos que hay en el agua, lo que facilita la navegación en aguas profundas y de poca visibilidad, a este proceso se le conoce como electrolocalización.

Desde el punto de vista de un pez eléctrico los objetos en general se podrían dividir en conductores y no conductores. En el caso de los conductores se cuentan los seres vivos y en el de los no conductores cualquier otro objeto inanimado como por ejemplo una roca.

Tanto unos como otros proyectan sobre la superficie electrorreceptiva del pez una imagen o "sombra" que varía en función del tamaño, la distancia, la forma e incluso el material del objeto, así como la morfología del órgano eléctrico y el cuerpo del pez [32].

Los elementos conductores son producto de una mayor concentración de voltaje mientras que los no conductores concentran menos voltaje, aunque sus bordes sí que lo concentran definiendo con mayor exactitud la imagen eléctrica que el pez recibe.

Otra de las principales funciones que cumplen estos electrorreceptores es la comunicación entre individuos que pertenecen a una misma especie, a este proceso se le conoce como electrocomunicación.

Estos individuos establecen un canal de comunicación privado entre dos o más organismos biológicos. Durante el proceso comunicativo la forma de la onda transmitida por el órgano eléctrico de cada individuo permanece constante, sin embargo, los patrones de intervalos entre pulsos (IPI) varían en función de la situación a la que se esté enfrentando el pez, es decir, variando el tiempo que pasa entre que emiten un pulso y el siguiente, de esta forma es cómo el pez establece la comunicación, por lo que estos intervalos serán distintos si el organismo está en fase de cortejo, está marcando territorio o está en una fase de ataque [37-43].

Durante la comunicación, cada individuo que participa tiene dos medios para obtener información, su propio campo eléctrico distorsionado por la resistividad de los elementos que hay a su alrededor y una corriente que capta de los otros sistemas con los que está interactuando.

El *Gnathonemus Petersii* posee dos fóveas electrorreceptoras en su *Schnauzenorgan* y su región nasal, las cuales son muy parecidas a la visual de la retina de muchos animales. Existen experimentos que han demostrado que esta especie de mórmido puede determinar los componentes resistivos y capacitivos de la impedancia de un objeto para por ejemplo identificar presas cuando está buscando alimento. Además, es capaz de medir la distancia y la forma tridimensional de los objetos mediante la electrolocalización activa (electrorrecepción activa). Este tipo de peces pueden detectar una serie de parámetros con los que forman la imagen eléctrica de un objeto, la amplitud máxima, la pendiente máxima, el ancho de la imagen y las distorsiones de la forma de la onda [32].

K. Diseño de experimentos software

Una de las ventajas que tiene RTXI es la gran facilidad con la que se pueden configurar los experimentos (Figura 35) y estudios gracias a la interfaz gráfica tan cómoda y sencilla que tiene. En este proyecto para configurar los experimentos hay que realizar los siguientes pasos.

Primero se cargarán todos los módulos desarrollados por el usuario y se modificarán los parámetros que forman parte de la interfaz, cuando ya se haya especificado el valor de todos los parámetros en cada uno de los módulos se seleccionará el botón *"Modify",* de esta forma todos los cambios serán comunicados a la parte en tiempo real de la tarea.

Después, haciendo uso del módulo ***Sync*** proporcionado por el código del sistema de RTXI, se seleccionarán aquellos módulos que formarán parte del experimento ya cargados diferenciándolos por su ID, de esta forma se sincronizará el inicio y la detención del conjunto de módulos estableciendo así el tiempo de duración del experimento. Por otro lado, este módulo permite sincronizar el proceso con el *Data Recorder*, módulo que nos aporta toda la información que se precise en tiempo real y necesaria para el estudio y el análisis de los resultados obtenidos en el experimento. Además, podemos hacer uso del *Data Recorder* para el depurado de funcionalidad mediante el grabado de señales y parámetros de los módulos que se han desarrollado.

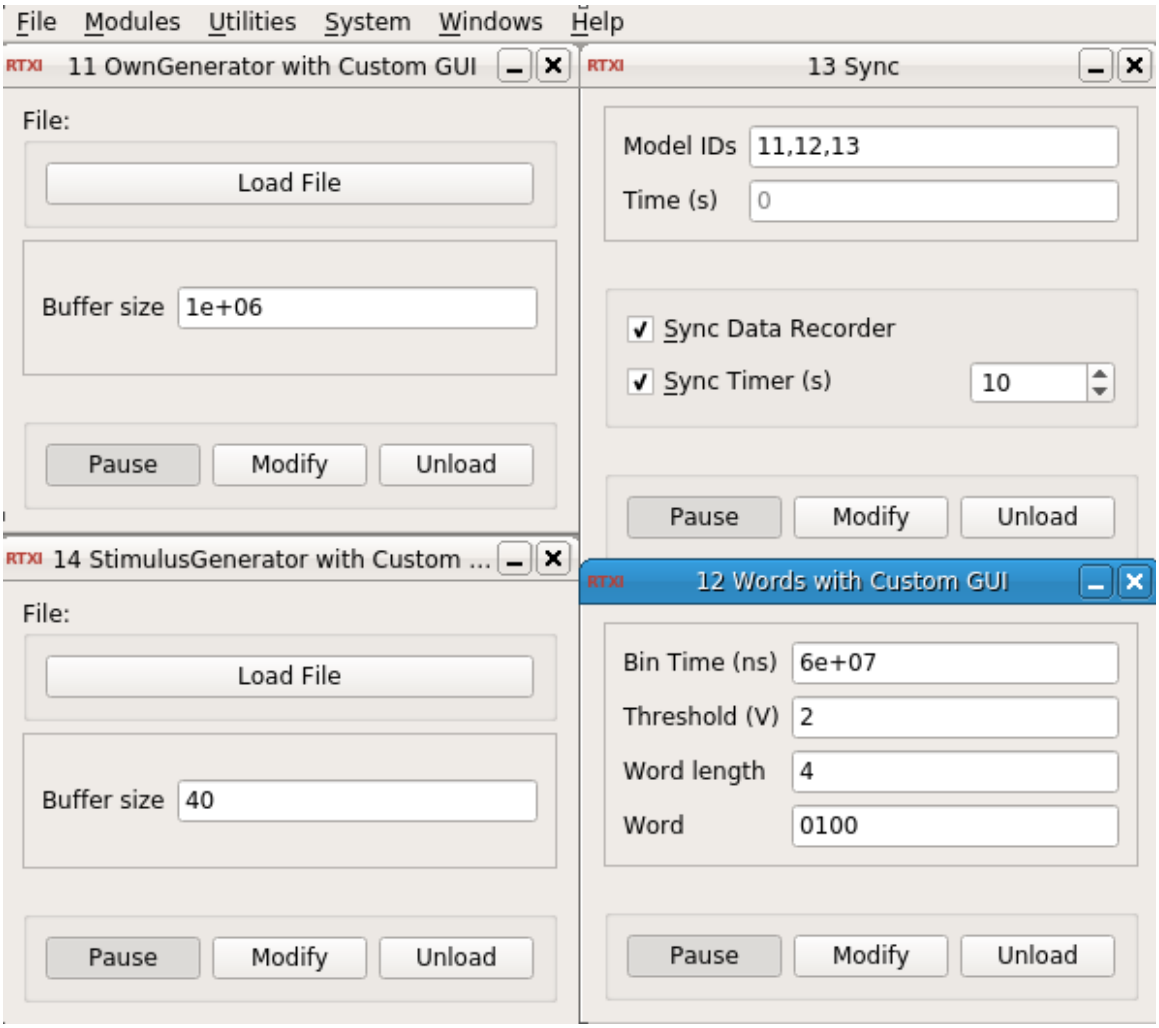

Figura 35 Configuración experimento.

A continuación, se configurará el *Data Recorder* (Figura 36) seleccionando las señales que se quieren grabar durante la ejecución de las tareas en tiempo real, como, por ejemplo, el voltaje del estímulo de salida del experimento, el de entrada, si se ha detectado o no una palabra, etc.

También se deberá seleccionar el archivo en el que se grabarán estas señales en formato HDF5.

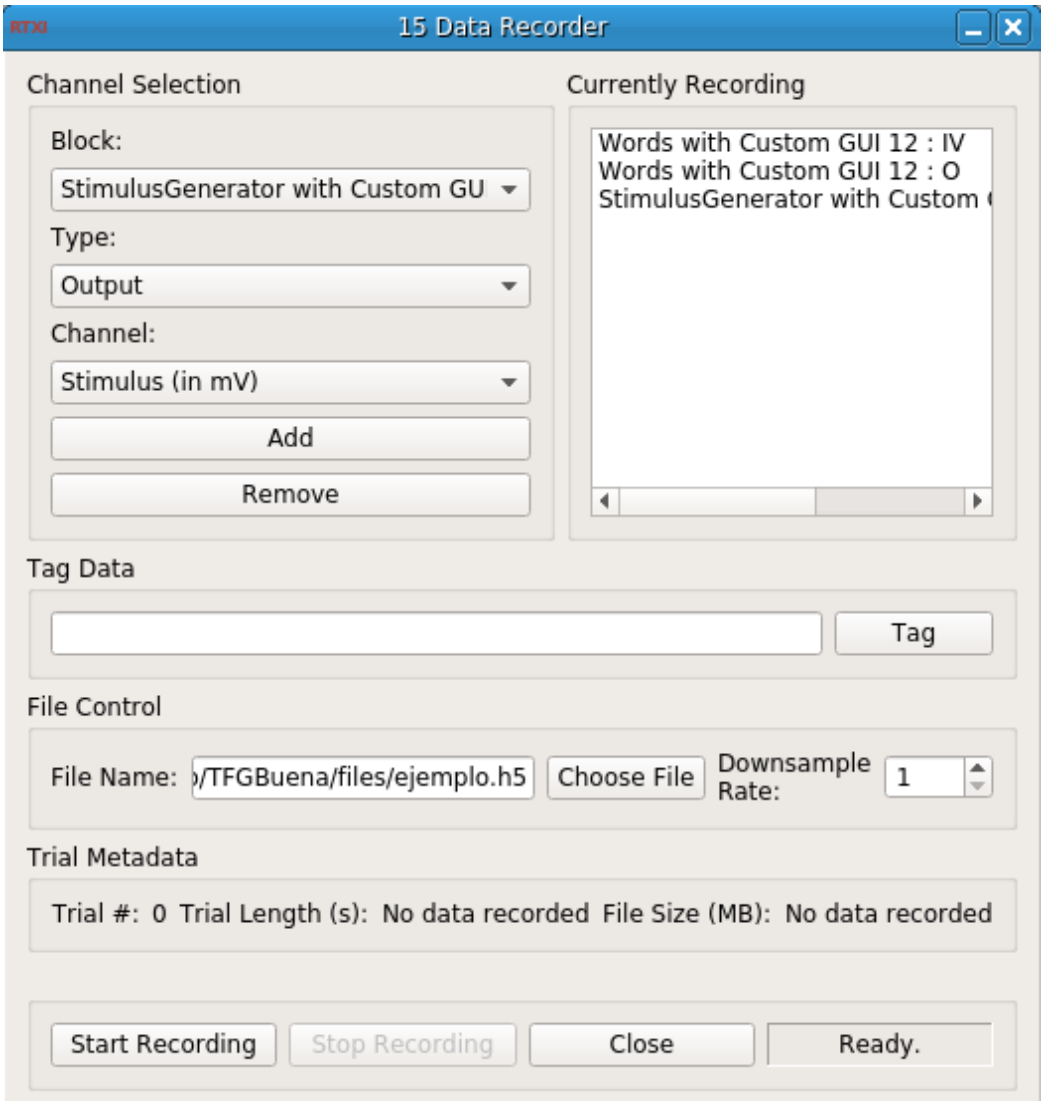

Figura 36 Interfaz *Data Recorder.*

Por último, para que el experimento funcione correctamente se deberán conectar los flujos de datos entre las entradas y salidas de los módulos desarrollados y que sean necesarias para el intercabio de información entre ellos, para ello se usará el módulo Conector (Figura 37) donde podremos seleccionar los módulos correspondientes así como sus canales de entrada y salida, una vez seleccionados se pulsará el botón *"Connect"* y la conexión quedará establecida.

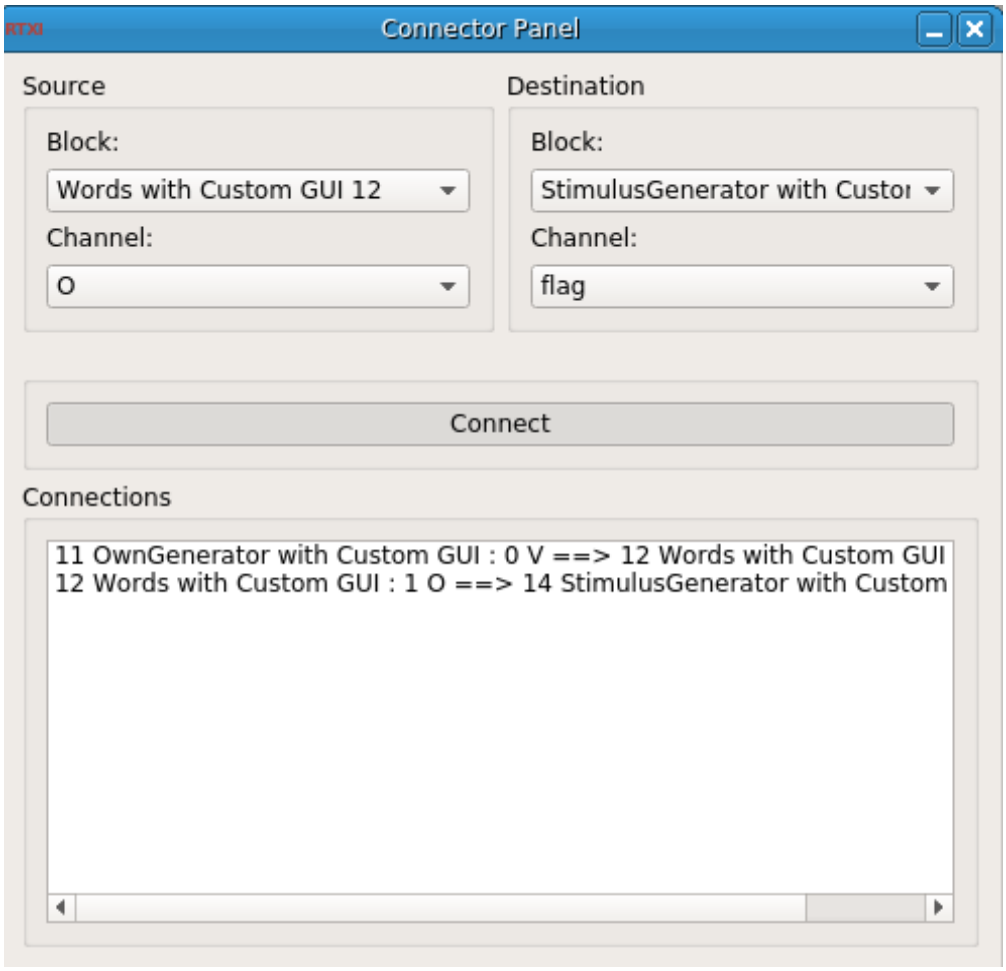

Figura 37 Interfaz *Connector*.

Una vez finalizada la configuración del proyecto se pulsará el botón *"Pause"* del módulo *Sync* dando comienzo a la ejecución en tiempo real del proyecto. Esta ejecución finalizará cuando acabe el temporizador establecido o se pause el módulo *Sync*, cualquier pausa de cualquiera de los otros módulos puede provocar la obtención de datos no deseados. Además, durante la ejecución se podrá cambiar cualquiera de los parámetros modificables de los módulos a traves de la interfaz gráfica, RTXI se encargará de comunicar estos cambios a los módulos mientras la ejecución sigue adelante.

Para finalizar es importante conocer algunas opciones que nos proporciona RTXI y que son de gran utilidad a la hora de realizar experimentos. Por un lado, esta herramienta cuenta con un panel de control (Figura 38) donde se puede cambiar la frecuencia/periodo del sistema, es decir, se puede modificar el espacio de tiempo en el que las tareas en tiempo real se ejecutan. Cuanto menor es el periodo, más cantidad de mediciones se pueden realizar y por tanto obtendremos datos más precisos.

Por otro lado, el panel de control permite seleccionar las tarjetas de adquisisción de datos de las que se va a leer en los experimentos, en este proyecto se ha trabajado con módulos de usuario y de sistema que permiten la generación de pulsos, sin embargo, gracias a la

interfaz de Analogy, esta herramienta es compatible con una gran cantidad de tarjetas de adquisición de datos que se pueden usar durante los experimentos.

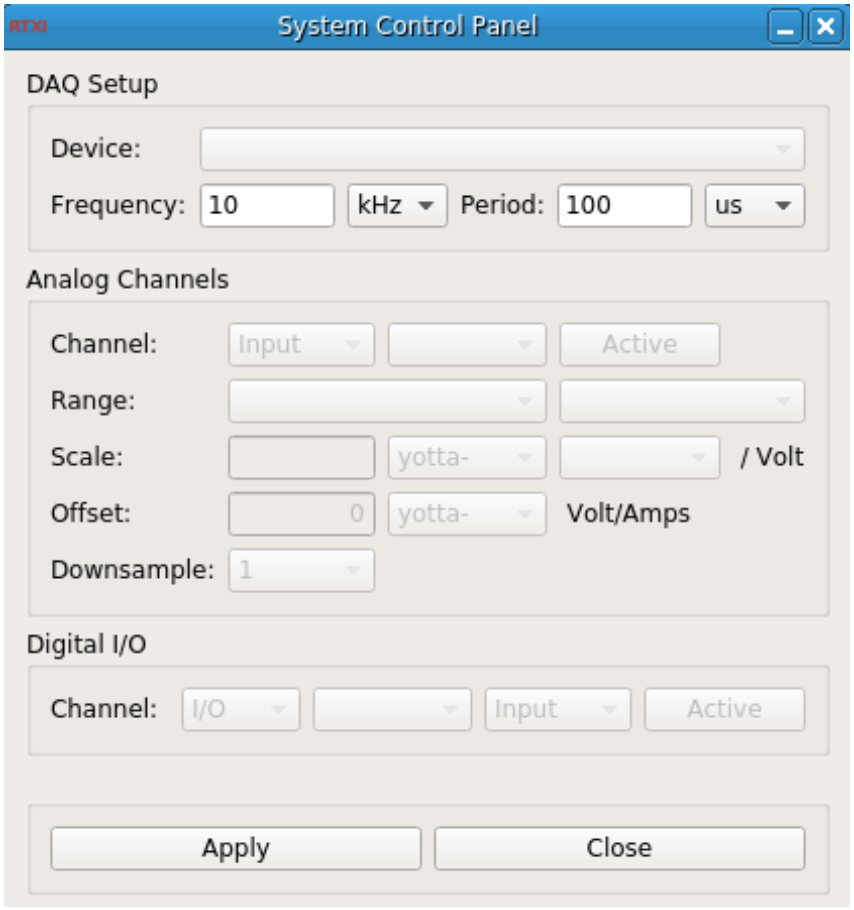

Figura 38 Interfaz *System Control Panel.*